Percolating Atomic Switch Networks for Neuromorphic Computing

A thesis submitted in partial fulfilment
of the requirements for the Degree of

Doctor of Philosophy in Physics
at the
University of Canterbury

by

Joshua B. Mallinson

School of Physical and Chemical Sciences

University of Canterbury

2020

Abstract

Conventional computer technology is ubiquitous in modern society. Initially used only to perform mathematical calculations, computers are now integrated with most daily human activity. As the demand on computers shifts towards more and more complex tasks, the limitations of conventional computing architectures are being realized. Conversely, many such tasks can easily be carried out by the human brain, which is capable of abstraction, recognition, prediction and decision making with extremely low power consumption.

The vast capabilities of the brain have motivated the development of novel neuromorphic computing architectures which can naturally facilitate brain-like computation. Several such architectures are being developed internationally, but few aim to mimic the complex connectivity and critical dynamics of the cortex; features which are central to the brain's computational power.

This thesis investigates percolating atomic switch networks (PASNs) as candidates for novel neuromorphic architectures. It is demonstrated that PASNs exhibit many of the same emergent phenomena as the brain, including scale-free temporal dynamics and long-range temporal correlations. It is further shown that the critical avalanche dynamics observed in neuronal firing patterns is also present in the sequences of switching activity exhibited by PASNs. Scaling exponent relations are used to definitively demonstrate that, like the brain, PASNs are critical systems.

Reservoir computing (RC) is a novel brain-inspired computing paradigm that utilizes temporal correlations to perform classification and prediction tasks. The nonlinear dynamical response of PASNs to electrical stimulation makes them particularly suitable candidates for physical reservoirs in neuromorphic RC systems. PASNs are thus utilized as the dynamical nodes in a time-delay RC framework. Through numerical simulations and experiments, it is demonstrated that PASNs can perform waveform classification and time-series prediction tasks, marking the first ever implementation of a PASN as a neuromorphic device.

Contents

Table of Figures

List of Tables

Acknowledgements

I am forever grateful to the plethora of people who have helped me along my PhD journey, without whom I could not have reached this point. I would like to thank Professor Simon Brown for the opportunity to work on this project and for the continued support and mentorship. I really could not imagine having a better supervisor. To Dr. Saurabh K. Bose, many thanks for taking me under your wing, and for all the help you have given me over the years.

I would also like to thank my fellow "Neuromorphers" who battled alongside me during this experience: Dr. Shota Shirai, Dr. Susant Acharya and Edoardo Galli. You all made my research possible. And my other cohorts, Dr. Maxime Lester, Dr. Tobias Maerkl and Sara Salehi for providing pleasant lunchtime discussions.

Further thanks to our collaborators Professor Phil Bones, Dr. Steve Weddell, Associate Professor Matt Arnold, Matt Pike, Dr. Kourosh Neshatian for helpful and intriguing discussions.

To Stephen Hemmingsen, Robert Thirkettle, Geoff Graham and Dr. Orlon Petterson, the services that you provide were essential to me achieving my goals.

Thank you to the University of Canterbury, the MacDiarmid Institute of Advanced Materials and Nanotechnology and to the Ministry of Business Innovation and Employment for facilitating my research. Special thanks to Professor Mike Reid and Professor Jenni Adams for your continued role in my education.

On a more personal note, Michelle and Peter Hancox, my Mum and Dad, I owe you more than anyone. Thank you for the life you have given me and for sparking my curiosity from a young age. To my father Brent Mallinson, thank you for your support and encouragement. To my friend Scott Camp, you believed in me long before I ever did, and I honestly would not have even begun studying physics if not for your influence.

And to all of those I have omitted: thank you!

Chapter 1

Introduction

In the last century, the vast power and utility of information systems has been harnessed by humanity. The invention of the solid state transistor in 1947, by John Bardeen, Walter Brattain, and William Shockley at Bell Labs, marked the beginning of the digital revolution, and the dawn of the information age [1]. The microminiaturization of transistors, and the development of semiconductor fabrication techniques, lead to the realization of the integrated circuit (IC) in 1959 [2]. Since then, the capacity and operational speed of ICs has progressed enormously as a result of decades of innovation aimed towards reducing transistor dimensions, and hence increasing the transistor density of ICs. This progression was encapsulated by Gordon Moore, and famously referred to as Moore's law: The number of transistors in a dense IC doubles approximately every two years [3]. What began as the observation of a simple trend quickly became a benchmark for the semiconductor industry, and for over forty years, Moore's law held fast. But with exponentially increasing transistor density, comes the exponentially increasing cost of the research and development required for each new generation of ICs [4]. Additionally, the size of individual transistors is approaching physical limitations: transistors on the latest generation of commercially available chips are as small as 7 nm [5,6]. In fact, Ref. [7] demonstrates a working transistor

comprised of a single atom, which is the absolute limit of miniaturization. These factors signify an imminent departure from Moore's law, which Gordon Moore himself has predicted will occur within a decade [8].

In order to sustain the rate of increase of computational power, future devices will need to adopt revolutionary new architectures. If the components of ICs cannot continue to be made smaller, consideration must be given to alternatives to both the components themselves, and to how these components are arranged. One such approach which has gained significant interest since its conception by Carver Mead in the late 1980s, is neuromorphic computing [9,10]. When it comes to abstractive computational tasks, such as pattern recognition, navigation, and decision making, the human brain surpasses even the world's best supercomputers, at a fraction of the energy consumption [10,11]. Neuromorphic architectures are inspired by the brain's ability to process a multitude of poorly conditioned sensory information and identify and extract the useful patterns and trends. Neuromorphic architectures use components which have similar functionalities to the neurons and synapses in the brain [12]. However, it has been pointed out that few of these approaches aim to mimic the complex and critical connectivity of such components within the brain [13], features which some believe lie at the heart of the brain's computational power.

Percolating atomic switch networks (PASNs) [14-19] , which are the focus of this thesis, have been shown to possess some desirable attributes of a neuromorphic system. They respond to electrical stimulus with complex conductance switching dynamics which may be utilized to perform computational tasks. Furthermore, PASNs are self-organised devices poised at criticality which are fabricated from metal nanoparticles (NPs) using scalable low-cost procedures.

The remainder of this chapter gives an overview of the mammalian brain (Section 1.1), followed by a description of artificial neural networks (Section 1.2) and neuromorphic computing (Section 1.3). Section 1.4 then introduces PASNs and describes the physical mechanisms by which they operate.

1.1 The Brain: A Biological Computer

The brain is a biological information processing system which is present in all vertebrate and most invertebrate animals. It is the primary component of the nervous system and has centralized control over a body's other organs. A brain is responsible for processing

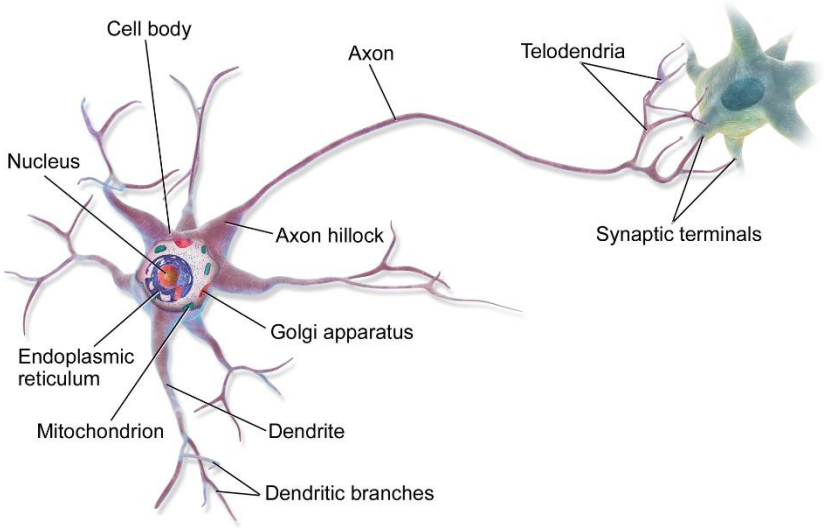
information collected by the sense organs (eyes, ears etc.) and subsequently driving the function of the organism to sustain its life. Raw sensory information is combined with information about the current needs of the animal and its memory of past circumstances (perception) in order to respond to its environment such that those needs are met (control). The exertion of coherent control over the actions of an organism requires the interplay of a large number of functional sub-systems. A centralized brain allows the integration of information, thus providing coordinated motor control, preventing the co-occurrence of interfering functions, and allowing stimulus in one area of the body to excite responses in another areas [20]. The result is an organism which can exhibit rapid and coordinated responses to changes in its environment.

The brain receives and transmits information throughout the body via both electrical and chemical signalling. The fundamental units of the nervous system are called *neurons*: cells which are capable of sending electrochemical pulses called *action potentials* over long distances to other cells via protoplasmic fibres called *axons* [21]. Figure 1.1 (a) shows a schematic diagram of a neuron cell. Neurons receive action potentials from other neurons through chemical junctions called *synapses*, and a single axon can have several thousand synaptic connections to other cells [21]. Synapses are the key functional elements of the brain because they regulate the connectivity between different neurons and thereby control how they communicate with one another. Figure 1.1 (b) shows a schematic diagram of a chemical synapse. When an action potential is emitted from the nucleus of a neuron (called the presynaptic neuron), it travels along the axon and arrives at a synaptic terminal where it triggers the release of chemicals called *neurotransmitters*. Neurotransmitters flow into the synaptic cleft and bind to receptors on the membrane of the post-synaptic neuron or target cell. Depending on the type of neurotransmitter released, the synaptic response may either be inhibitory or excitory. The post-synaptic neuron has synaptic connections to many other presynaptic neurons and therefore receives a multitude of both inhibitory and excitory signals. If the sum of excitory signals being received exceeds the sum of inhibitory signals by some threshold, then the neuron may release its own action potential. The firing neuron then becomes the "presynaptic" neuron and the signalling process repeats with new target, or "post-synaptic" neurons [22].

The neurons within the brain are highly interconnected by synaptic junctions forming a complex network. Which neurons are firing at any given time is determined not only by the input signals from sensory organs, but also by which neurons were firing at some previous

time, and by the relative strength of the synaptic connections between the neurons. Repeated exposure to the same sensory input can result in a strengthening (or weakening) of synaptic connections, leading to stronger excitory (or inhibitory) synaptic responses. This can increase (or decrease) the likelihood of post-synaptic neurons firing, resulting in a different response from the network of neurons to the sensory input. This is called *synaptic plasticity* and is thought to be the primary mechanism responsible for learning and memory in the brain [21,23]. Learning is an emergent property of the complex network of interacting neurons: learning cannot be achieved by individual neurons as it requires adaption of the synaptic connections *between* the neurons. Individual neurons themselves, while crucial to the

(a)



(b)

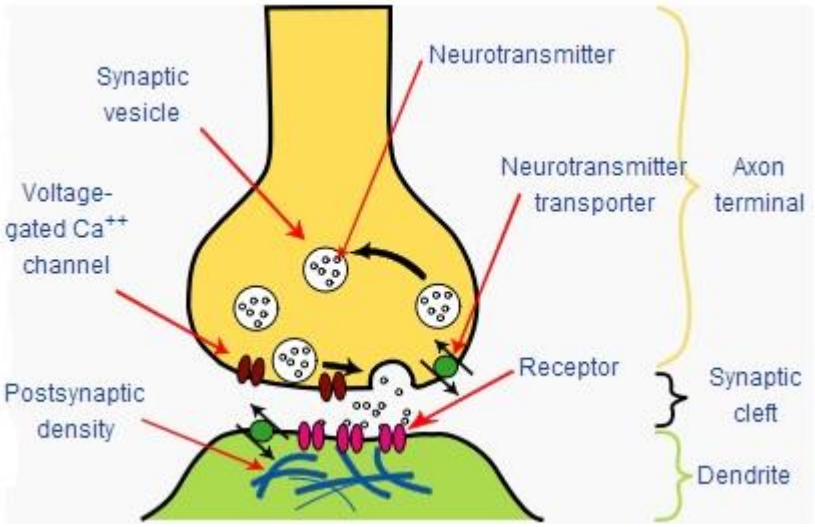


Figure 1.1: Neuron and synapse schematics. (a) Schematic diagram of a neuron cell. Action potentials are generated in the nucleus and transmitted to other neurons via synaptic terminals. (b) Schematic diagram of a chemical synapse. Action potentials from the presynaptic cell trigger the release of neurotransmitters from the axon terminal which are received by the dendrite of the post-synaptic neuron. Reproduced under creative commons licence (CCL): Wikipedia / CC BY-SA (https://creativecommons.org/licenses/by-sa/4.0).

function of the nervous system, are relatively simple entities: if they are sufficiently stimulated, they release an electrical pulse which contributes to the stimulation of other neurons. It is the topology of the network, which is modulated by synaptic connections, that is responsible for *how* neurons affect each other, and for how the state of the *entire network* of neurons evolves in time [24]. While the detailed mechanisms of individual neurons and synapses have been elucidated by extensive studies, the manifestation of intricate and complex behaviours from the interaction between ensembles of billions of neurons is yet to be fully understood [25].

The structure of the brain, referring to the topology of the connections between all of the neurons, is neither regular nor random, but somewhere in between. The networks within the brain have been shown to be highly clustered like a regular lattice while also having a short path length like a random graph [26]. This is called a small-world network topology and is thought to provide the brain with a host of computational benefits such as high efficiency and minimal wiring cost while maintaining a high degree of global connectivity [27]. It has also been shown that the connectivity in cortical networks is fractal [28] and scale-free [29], meaning that connection patterns repeat on smaller and smaller scales, and that some neurons which are connected to huge numbers of other neurons, act as main hubs for information transfer. It is thought that this brain structure was selected by evolution to enable the brain to carry out adaptive functions, like learning and pattern identification, efficiently and for minimal evolutionary cost [26].

The information processing in the brain is spatio-temporal. It depends on how the complex network of billions of neurons and trillions of synapses evolves in time, with the state of each neuron being affected by other neurons and by its own history. Together, the interconnected neurons form a chaotic system which is strictly deterministic, but which is so complex and nonlinear that its exact state at some point in the future is impossible to predict. This type of behaviour is typical of systems which operate in the vicinity of a critical point; often termed ‘at the edge of chaos’. In fact, considerable evidence has accrued that the brain indeed operates near a critical point and is a self-organised critical system [30-33]. This is characterized by bursts of activity amongst neuronal populations called *neuronal avalanches*. Critical systems are sensitive to small perturbations which provide the brain high dynamic range and responsiveness, allowing adaptive behaviour. It is hypothesized that criticality bestows the brain a number of other functional advantages such as maximum computational performance [13,33-36].

Animals require the ability to respond quickly, adequately, and efficiently to changes in their environment in order to be selected by evolution. This response requires the interplay of a large number of functional subsystems, and the real-time interpretation of incoming sensory information. To achieve these tasks in the most efficient way, nature produced the brain, a highly complex network of nonlinearly interacting neurons and synapses, to control the animal's response to its environment. When it comes to tasks like real-time processing of images, patterns, audio, and abstractive tasks like classification and decision making, there is no better, and certainly no more efficient computer in the world than the human brain. However, information processing in the brain is very different to information processing in conventional computer chips (Section 1.3.1) where the networks of transistors are highly regular arrays and the behaviour of the elements are well defined and instruction based. Thus, neuromorphic architectures are required to harness the computational power and efficiency of the biological brain.

1.2 Artificial Neural Networks

Artificial neural networks (ANNs) [37,38] are software implementations of bio-inspired computing principles. They utilize some of the functionality of cortical networks to varying degrees of bio-realism, but in all cases, ANNs are a simplification of the immensely complex and intricate biological brain. ANNs utilize a collection of interconnected nodes, which are analogous to neurons, with weighted connections that are analogous to synapses. Nodes receive input from other nodes via these weighted connections, and based on that input, transmit signals to other nodes. This is a simplified version of the operational principles of the brain described in Section 1.1. Like the brain, ANNs perform tasks by *learning*, rather than by following predefined instructions. For example, an ANN may learn to classify bananas upon being presented with images containing different fruit. The ANN must first be told which images contain bananas (training) and then can perform the task without being told which images contain bananas (classification). The ANN performs this task without knowledge of the features of a banana; that they are slender and curved, or that they are yellow, for example. Instead, ANNs identify distinct features of the examples they are shown automatically. They are not looking for a yellow curved blob in the image, but rather looking for identifying features which were present in previous examples of images of bananas. In

contrast, a conventional computer algorithm may investigate a series of logical pre-scripted conditions: does the image contain a yellow region? If so, is the aspect ratio of the yellow region greater than some threshold? If so, … etc. until the algorithm arrives at a conclusion about whether the image contains a banana. Investigating many conditions for many different classes (apples, watermelon etc.) requires a huge number of logic operations to produce a single classification. ANNs on the other hand learn the features of the different classes and can identify them at once without being told to look for certain attributes of that class, resulting in excellent performance for classification and regression tasks.

ANNs are networks of nodes which interact via weighted connections. A weighted connection is simply a pathway for information where the flow of the information can be high for a strong connection (large weight) or low for a weak connection (small weight). In training an ANN, learning is achieved by adjusting the weights of the connections between nodes such that the network response is closer to the desired response. For example, imagine that images of fruit are used to train an ANN to identify bananas. If the ANN successfully classifies images as either containing (or not containing) bananas for some fraction of the examples it is shown, then the connection weights are systematically altered so that the fraction of successful classification increases. This can be repeated iteratively to achieve the maximum possible number of successful classifications.

There are many different types of ANN models of varying complexity, and they typically have their own training algorithms. All ANNs however are based on networks of nodes, and all training algorithms involve correctly choosing the weights between them. More complex models typically have greater functionality/better performance, but also incur greater training and operating costs. The following sub-sections introduce some of the key ANN archetypes.

1.2.1 Feed-forward Neural Networks

The nodes in an ANN are typically organized into layers. ANNs in which information passes in one direction from layer to layer is called a *feed-forward neural network*, and the simplest illustration of such is the single-layer perceptron (SLP).

In its simplest form, an SLP contains a single artificial neuron which is represented by a threshold function. Figure 1.2 shows a schematic of a SLP and an equivalent biological analogy [39]. There are n biological neurons passing their signals x to a neuron via synaptic connections with weights w. In this example, the post-synaptic neuron is the perceptron and

the pre-synaptic neurons are input nodes. The function of the perceptron is to output a value of 1 if the weighted linear sum of the inputs (ξ) exceeds its own threshold value (b), and 0 otherwise. As a linear classifier, the binary outputs can be interpreted as either confirming or denying the membership of the input to a particular class. The perceptron can be trained to classify using a set of example inputs. Learning occurs by incrementally adjusting the weights (and threshold) so that the difference between the target output and the perceptron output is minimized [39]. As the error is a function of all of the weights, it forms a complex multidimensional hyperplane, and the goal of training is to find the global minima. There exists a training rule which will yield the optimal set of perceptron weights in a finite number

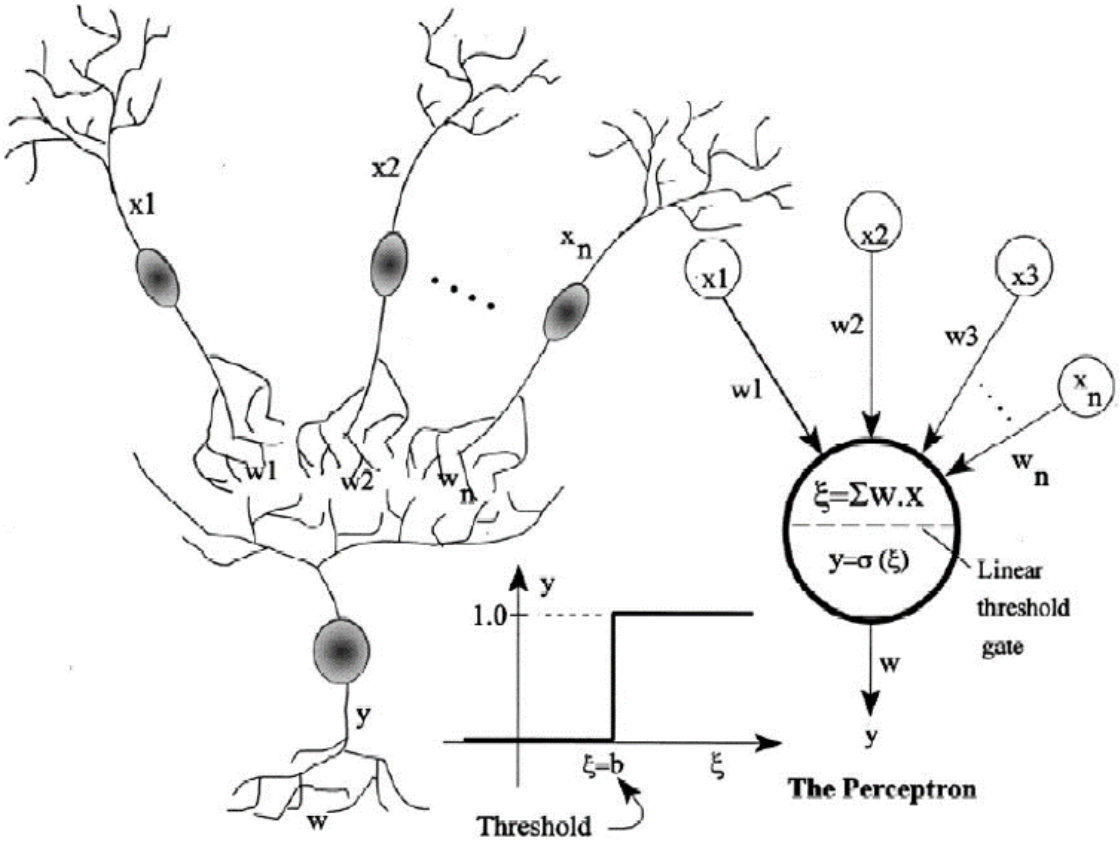


Figure 1.2: Schematic of a perceptron. A depiction of a biological neural network and the equivalent single-layer perceptron. n signals x_i are fed from other neurons (input nodes) via synaptic junctions (weighted connections, w_i). The "integrate-and-fire" action of a biological neuron is crudely approximated by a linear threshold function σ: if the weighted linear sum of inputs ξ exeeds the neuron's threshold potential b, then the output is one and the neuron is said to be "activated", otherwise the output is zero. As a linear classifier, the binary outputs can be interpreted as either confirming or denying the membership of the input to a particular class. Reproduced with permission from Ref [39]

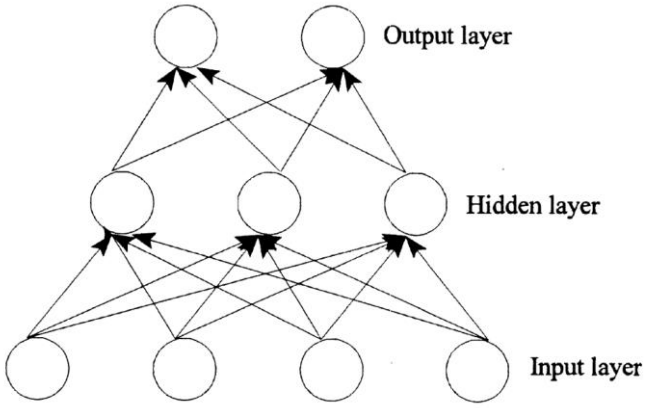


Figure 1.3 Schematic of a multi-layer perceptron. The addition of a "hidden layer" between the input and output layers allows the perceptron to be trained on nonlinearly separable classes. Reproduced with permission from Ref [41].

of iterations [40], but this rule can only perform accurately for classes which are linearly separable [41].

Classification of nonlinearly separable classes requires an additional layer of nodes between the input and output nodes, leading to the multi-layer perceptron (MLP) [41]. The schematic diagram in Figure 1.3 shows a MLP which features a *hidden layer* between the input and output layers. This layer of neurons is described as "hidden" because the states of the nodes in this layer are never examined: the inputs are fed to the input nodes and the output is taken from the output nodes, but the nodes in the hidden layer do their job (explained below) quietly in the background. This architecture serves as the basis for all other more complex ANNs which may have additional hidden layers (deep neural networks) or different connectivity structures (convolutional and/or recurrent neural networks).

The role of the hidden layer is geometrically illustrated in Figure 1.4 (taken from Ref. [38]). The SLP performs a linear classification by creating an $(N-1)$-dimensional hyperplane within the N-dimensional network space, where N is the number of inputs to the single layer. If the two classes are separable, but cannot be separated by a single hyperplane, the problem is said to be nonlinearly-separable. Such a problem requires a region of the N-dimensional space to be confined by multiple hyperplanes which can only be achieved by the addition of a second layer. A good example is the XOR problem which is illustrated in Figure 1.4. Each node in the SLP can draw a line through the two-dimensional (2D) space, but it requires a second layer to combine those lines to form the isolated region containing the black circles, just as XOR cannot be computed by any single Boolean logic operation. The problem requires two "layers" of logic operations: the first layer computes whether one element is true using OR (a linear operation) and also that both elements are not true using NAND (again, a

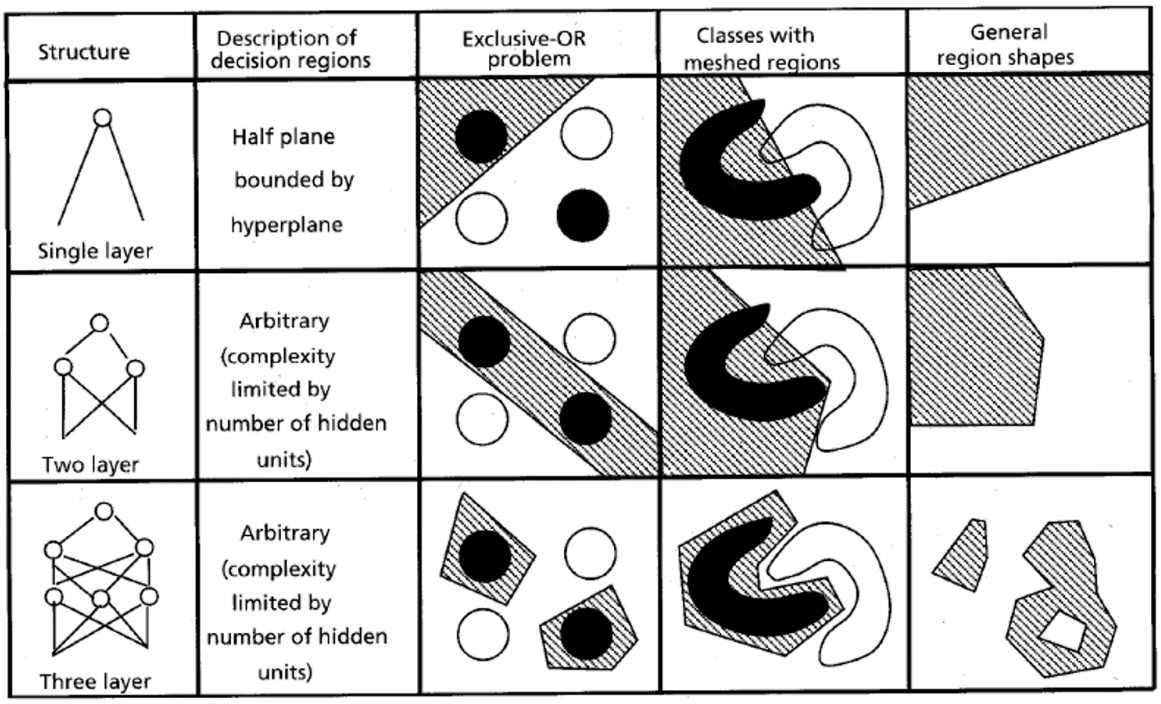


Figure 1.4: Geometric illustration of the role of the hidden layer in a MLP. SLPs can only separate classes using a single hyperplane. The hidden layer allows the composition of hyperplanes so that nonlinear boundaries can be formed. The universal approximation theorem states that a MLP can approximate any function, meaning that it can create an arbitrarily complex boundary, given it has sufficient nodes. Reproduced with permission from Ref. [38].

linear operation). The second layer computes its own linear operation using the outputs of the first layer as the input: it checks that both of the previous conditions are true (using AND).

While the nodes in a feed-forward are not explicitly performing these logic operations, they are performing linear classifications, and the effect of layering those linear operations allows for nonlinear separation of different classes. In fact, the MLP has been shown to be a universal approximator: a multi-layer feed-forward network using a finite number of nodes and a logistic sigmoid activation function is theoretically capable of approximating any continuous function [42]. It has further been shown that this rule holds for any non-polynomial activation function [43]. The caveat is that the so-called *universal approximation theorem* does not address the learnability of the required parameters: while it may be in theory possible for a MLP to learn "anything", in practise it is often a matter of overcoming other constraints such as computation time, the vanishing gradient problem [44], or getting stuck in local minima during training [45]. Hence there is motivation for more complex architectures which can mitigate these practical barriers, despite the universal approximation theorem stating that a single hidden layer is sufficient.

Multi-layer feed-forward networks can use a variety of training methods to optimize their weights. The most popular of these is *back-propagation* where the weights are initialized to

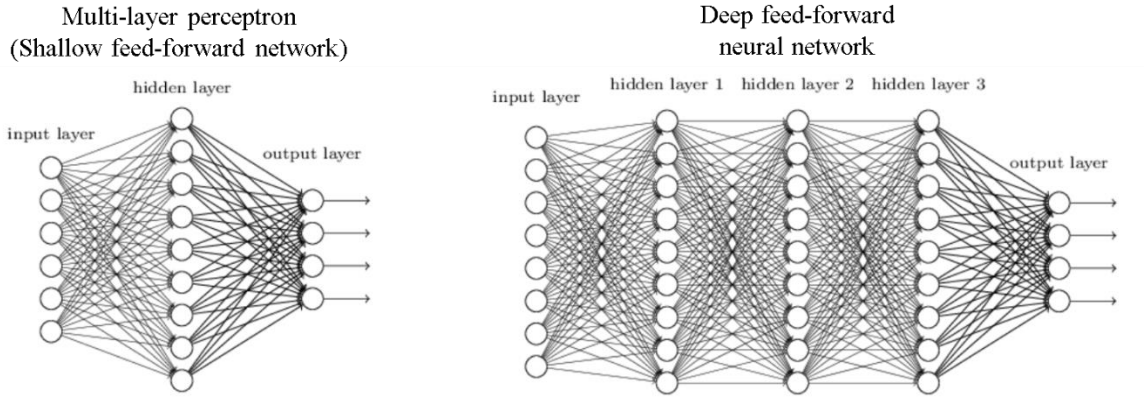


Figure 1.5. Schematic of a deep neural network. The addition of multiple hidden layers to a MLP results in a deep neural network (DNN) which are capable of modelling complex nonlinear relationships due to the additional layers of abstraction. Reproduced with permission from Ref [48].

random values and iteratively updated in response to feedback of the calculated error between the target and the network response. This process, called *gradient decent*, involves calculating the derivative of the error function with respect to the weights, and then adjusting the weights such that the error decreases [46]. Gradient decent, and therefore back-propagation, thus require activation functions to be continuously differentiable. For this reason, the stepwise threshold function σ in Figure 1.2 is typically replaced by a logistic sigmoid function, though other functions are also used [47].

A *deep neural network* (DNN) is an ANN which features multiple hidden layers between the input and output layers [49], as depicted in Figure 1.5. Deep neural networks are capable of solving complex problems because the primary operations performed by the lower layers (e.g. identifying features of a class) are composited by the higher layers. Furthermore, it has been demonstrated that DNNs can model complex nonlinear relationships using fewer nodes than a similarly performing shallow ANN [49], and that adding more layers to a DNN can increase its performance, as depicted in Figure 1.6 (taken from Ref. [50]).

One consequence of the many layers of abstraction mean that the DNN is very good at picking up obscure trends within the data; some of which may not be the desired trend. This is called *overfitting* (discussed in Section 4.3.5) and is usually mitigated by regularization techniques [51,52]. *Convolutional neural networks* (CNNs) [53] incorporate such regularization measures into their architecture. The MLP and DNN are fully-connected, meaning that each node in one layer is connected to every node in the next layer, but the CNN utilizes a hierarchical connectivity and shared weights in order to assemble complex patterns from the combination of smaller, simpler patterns. They typically feature at least one "convolutional" layer which uses convolution in place of matrix multiplication, as well as a

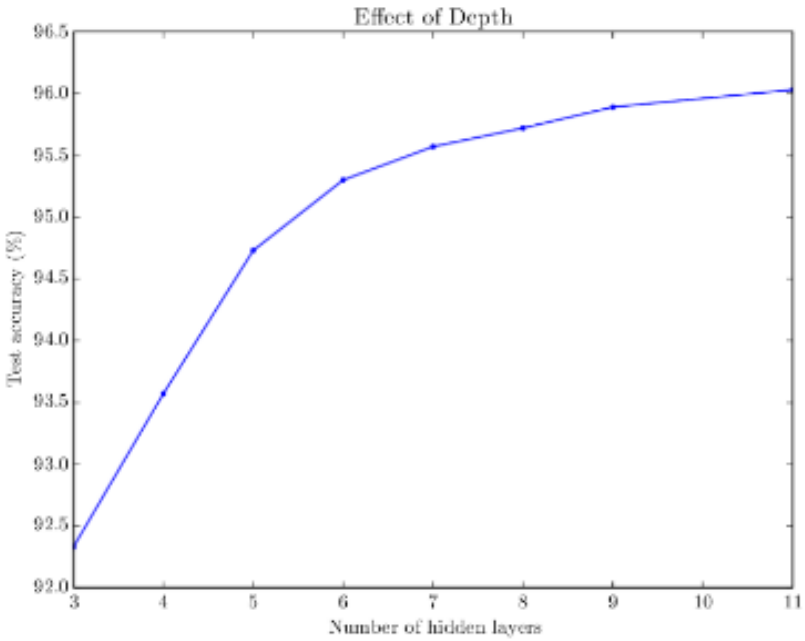


Figure 1.6. Effect of depth of a DNN. Empirical data showing that the greater the number of layers in a DNN, the better its performance at transcribing multi-digit numbers from photographs of addresses. Reproduced with permission from Ref. [50].

host of other different types of layers such as pooling layers, fully connected layers, ReLU layers, and normalization layers. While the details of this architecture are beyond the scope of this study, keen readers can find further information in Ref. [54] and citations therein.

1.2.2 Recurrent Neural Networks

Section 1.2.1 introduced several models of feed-forward ANNs where information is always passed in one direction. This section expands this discussion to include ANN models that feature *recurrent connectivity* i.e. pathways which form loops within the network. The majority of the information presented here can be found in the review [55] and its citations.

Feed-forward ANNs perform very well at abstraction and classification tasks. Examples presented to the network during training invoke some response from the network which is mapped to the desired outcome by adapting the weights representing the connections between the nodes. This network response depends only on the example being presented to it – not on any previous inputs, or any previous outputs, and once this example has been processed, the state of the network is lost. This is fine for tasks in which examples which are generated and processed independently, such as images of fruit which may contain bananas, but poses a serious problem if the examples being presented are related in time or space, such as in the case of time-series prediction [56] and sequence classification tasks like text recognition [57].

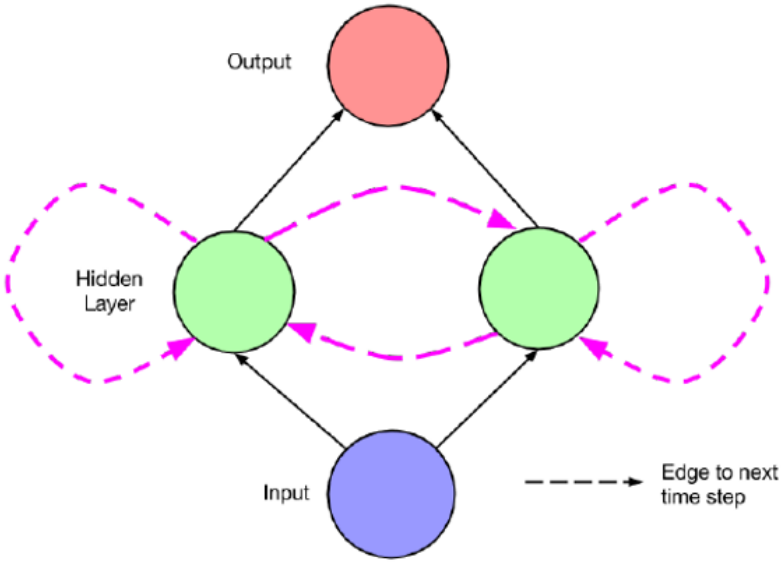


Figure 1.7: Schematic of a recurrent neural network (RNN). Augmenting a MLP by adding directed information pathways which span different time steps (dashed arrows) creates a RNN which features a fading memory of inputs and allows for successful classification of *sequences* of inputs, as required in tasks such as time series prediction and speech recognition [55].

Recurrent neural networks (RNNs) [58] feature directed information pathways which form recurrent loops. This endows the network with a fading memory of its past states: the inputs that each node receives is partially made up of the output of that (or other) nodes(s) at some previous time. This is illustrated schematically in Figure 1.7. Because of this fading memory, RNNs can process *sequences* of inputs and can identify patterns *in time*, which is extremely useful in real-world applications like stock forecasting and speech recognition. However, the additional recurrent connections are not without consequence: the training of RNNs is immensely complicated because some of the connections are passing information through time. For backpropagation, this requires that the recurrent networks be "unrolled" in time, as depicted in Figure 1.8. Each time step essentially becomes a layer in an unrolled non-recurrent network, meaning that one ends up training the equivalent of an extremely deep feed-forward neural network. The consequences of this training scheme are high computational and temporal cost, as well as exacerbation of the vanishing/exploding gradient problem [44]. Hence, there exists motivation for the exploration of alternative RNN models which exploit the power of recurrence while also providing a practically implementable training procedure.

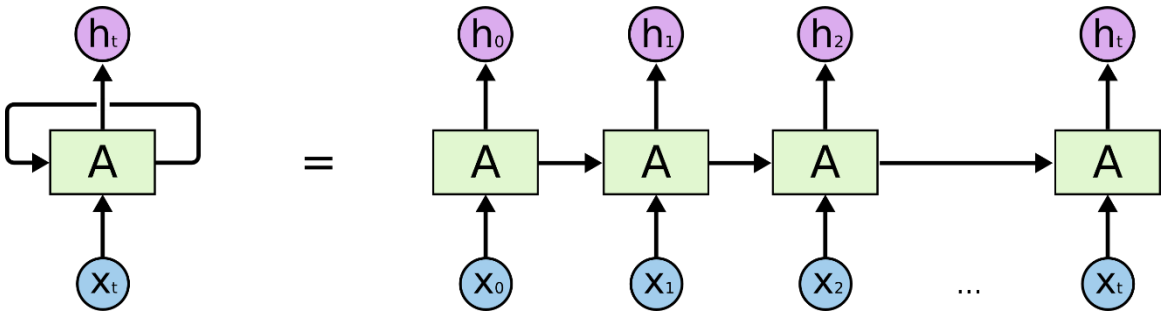


Figure 1.8: Unrolling a RNN in time. Schematic showing a recurrent node from within a RNN (left) and its equivalent form after unrolling in time (right). Each time step becomes a layer and the recurrent connection becomes feedforward connection, allowing for backpropagation through time. This diagram shows only one node, but this happens in parallel to all nodes in the network, meaning that it is a very complex and very deep feed-forward network which ends up being trained. Reproduced from Ref. [59] (permission requested).

1.2.3 Reservoir Computing

Section 1.2.2 introduced RNNs and described a very complex and computationally expensive adaptation of backpropagation which makes RNNs extremely difficult to train. This difficulty arises due to the huge number of weights which must be trained, both between nodes of the network, and between time steps. One approach which works surprisingly well is to simply not train all of the connections. If one were to randomly initialize all of the weights in a RNN and then only adapt (train) the weights in the final (output) layer, then the training procedure is reduced to a simple linear operation resulting in immense improvement in computational cost and training speed. This RNN model is called *reservoir computing* (RC) [60], because the hidden layers of the network act as a dynamic reservoir and the output layer maps the state of this reservoir to the desired outcome.

Two well-known realizations of RC are the echo state network (ESN) [58] and the liquid state machine (LSM) [61] which were developed independently during the early 2000s. The popularity of ESN and LSM frameworks at the time of their realization was due to the speed and ease of calculating the weights, compared to the challenge faced in adapting all connections as in most approaches to RNN training. With the advent of *deep learning* [62-64], these challenges have essentially been solved, and the appeal of RC software implementations is fading in favour of those more advanced RNN training techniques [65]. However, in a dynamical system where individual connection weights cannot be accessed directly, RC is an appealing method of achieving computation. Such dynamical systems may in fact be represented by physical reservoirs which are analogue to recurrent neural structures.

Several implementations of hardware-based RC have been realized using: functionalized nanowire networks [66,67], passive silicon photonic reservoirs [51,68], mechanical nano-oscillator systems [69], carbon nanotube/polymer networks [70], and memristor based neuromorphic microchips [71]. While these approaches are novel and have shown the ability for pattern recognition and time-series prediction, scalability is a concern when it comes to creating functional devices. Conversely, the low-cost fabrication of PASNs uses self-assembly of nanoparticles to provide a promising low-cost route to scalable RC hardware. Detailed discussion on RC, and particularly time-delay reservoir computing (TDRC), can be found in in Chapter 4.

1.3 Neuromorphic Computing

Neuromorphic computing is a concept popularized by Carver Mead in the late 1980s [9]. It describes how the implementation of very-large-scale-integration (VLSI) may be used to combine analogue circuits, in order to mimic the structure and electronic behaviour found in the biological nervous system [9]. In modern times, the definition has been extended to include VLSI of both analogue and digital circuitry, as well as software implementation of neural models, such as neural networks and deep learning algorithms. Software based implementations, while novel, still face the same imminent limitations as Moore's law: they rely on the complementary metal-oxide semiconductor (CMOS) transistor and the von Neumann architecture, the enhancement of which can no longer be sustained by miniaturization methods. Hence, this thesis is focused on the hardware implementations of neuromorphic computing, more rigidly called *neuromorphic architectures*.

1.3.1 Neuromorphic versus von Neumann Architectures

Since the realization of the first IC in 1959, computer processing power has increased in accordance with Moore's law [3]. This is predominantly due to the development of advanced semiconductor fabrication techniques, enabling more and more transistors to fit on a Si wafer of constant size. Because transistor dimensions are approaching physical limitations [4], and the cost of developing each new generation of ICs is increasing exponentially [6], new methods of increasing computational power must be sought. This section gives a brief

introduction to both the von Neumann and neuromorphic architectures and highlights some key differences between the two. More detailed discussions on the von Neumann and neuromorphic architectures can be found in Refs. [72,73], and Refs. [9,10] respectively.

Conventional computers rely on the *von Neumann architecture*, named for its principle inventor, mathematician and physicist John von Neumann (1945). This architecture features four primary components: the central processing unit (CPU), containing an arithmetic logic controller and processor registers, and a control unit containing an instruction register and program counter; a memory unit for storing data and instructions; a secondary storage unit such as a hard drive; and I/O ports for receiving and returning information. A schematic representation of the von Neumann architecture is shown in Figure 1.9 (a).

Computation is achieved when programmatic instructions, which are stored in the main memory, are sent to the CPU. The arithmetic logic controller executes the instructions, accessing data and additional program instructions from the main memory as required. Once an instruction has been executed, the generated output is returned to the main memory. During this process, the control unit is responsible for managing memory addresses for read and write operations, as well as controlling the flow of program instructions. Because the program instructions and the associated data are both stored in the main memory, they share a common bus to the CPU. Consequently, the CPU cannot conduct a read/write operation *and* fetch a program instruction simultaneously. In the early days of the von Neumann architecture, this was not an issue: main memory was relatively small and processor clock speeds relatively slow (kB and kHz respectively). However, modern computers feature CPUs with GHz clock speeds, and fast, random-access memory (RAM) containing GB of storage. As a result, the bus between the main memory unit and the CPU limits the data throughput to levels below the capacity of both the CPU and memory unit. This is known as the *von Neumann bottleneck*, and is a fundamental limitation of the architecture.

Registers and cache memory are small, fast access memory units which are situated very close to the processing core. Frequently used instructions or data can be stored in these units to reduce the latency introduced by the von Neumann bottleneck, but it cannot be eliminated entirely. Furthermore, as the capacity of processors and memory units increases, the worse this bottleneck becomes. Fast processors which are spending many cycles simply waiting for data transfer through the bottleneck costs both time and energy, resulting in computers which are less efficient.

Conventional computers, based on CMOS ICs and the von Neumann architecture, are irrefutably superior to the human brain at sequential, arithmetic calculations. For example, a simple calculator contains the most basic CMOS technology and is capable of multiplying numbers in a fraction of time it takes a human to do the same. However, if the calculator is tasked with recognising one face amongst a crowd of people, obviously it will fail, and even the world's best supercomputers would require megawatts of energy, and several seconds (or longer) to complete this task. The human brain however, easily completes the task in milliseconds, requiring only ~20 W of power [74]. The brain of the fruit fly is more efficient still: capable of intricate nonlinear tasks such as real-time flight control, predator avoidance, and the search for food and mates, all while consuming mere microwatts of power [75]. There is clearly a huge amount to be learned from nature regarding efficient computation, especially when it comes to nonlinear, abstractive tasks like decision making and pattern recognition.

Neuromorphic architectures [76,77] are based on the inspiration of biological neural systems. Rather than the centralized processing core and distinct memory unit of the von Neumann architecture, neuromorphic architectures feature highly distributed and parallel processing, with integrated memory. More specifically, a large number of artificial neurons are arranged in a complex interconnected network, where electrical signals are transmitted between them via weighted connections represented by artificial synapses. A schematic representation of this layout is depicted in Figure 1.9 (b). The inherently distributed nature of neuromorphic architectures enables highly parallel information processing and memory storage, thus avoiding the von Neumann bottleneck entirely. Memory is achieved by the

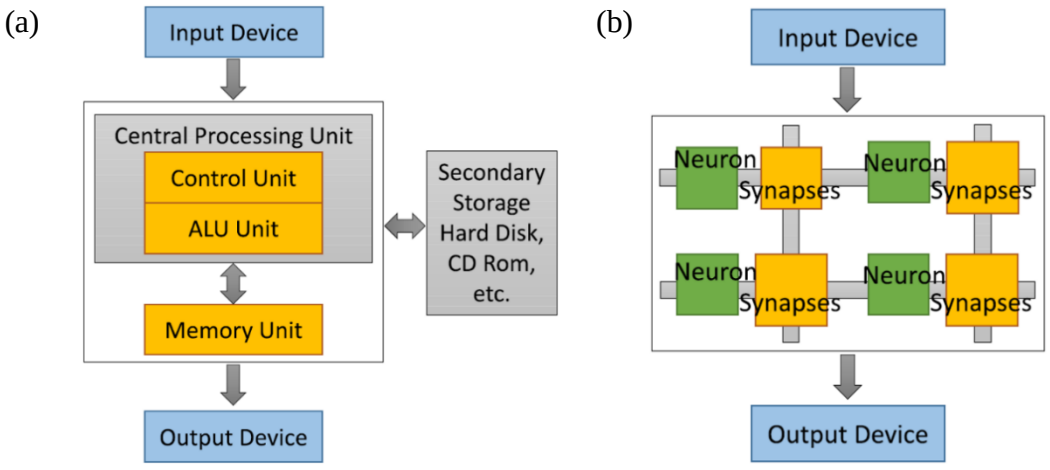


Figure 1.9: von Neumann vs Neuromorphic architectures. Schematic representations of (a) von Neumann architecture, and (b) neuromorphic architecture. Reproduced from Ref. [10].

plasticity of synaptic connections: the connection weights are updated in response to neuronal stimulus. This history dependent connectivity allows neuromorphic systems to learn new tasks and adapt to uncertain environments [78].

ANNs (Section 1.2) are software based approaches to neural computing which are implemented using standard CMOS technology and the von Neumann architecture. Thus, they enable conventional computers to perform tasks using bio-inspired computing principles. However, there is a fundamental incompatibility between the discrete state operation of conventional computing hardware and the brain-like computations which are forced to operate on it. This limits the capabilities of ANN approaches in software which require massive amounts of energy to perform tasks that the brain can perform with astonishing efficiency. Neuromorphic architectures are therefore required to provide a hardware platform that is naturally amenable to performing brain-like operations quickly and efficiently.

1.4 Percolating Atomic Switch Networks

This section introduces percolating atomic switch networks (PASNs) and several related concepts. Section 1.4.1 introduces nanoclusters and nanoparticles, giving a brief summary of PASN fabrication. Section 1.4.2 describes the coalescence of clusters after landing on the substrate, and Section 1.4.3 discusses percolation theory in the context of a network of conducting nanoparticles. The two operational regimes of PASNs are then introduced; the tunnelling regime in Section 1.4.4 and the switching regime in Section 1.4.5. Finally, Section 1.4.6 details the history dependence of the switching voltage threshold which separates the tunnelling and switching regimes.

1.4.1 Nanoclusters and Nanoparticles

Nanoclusters (or simply clusters) are particles comprised of groups of atoms which have at least one dimension in the 1 – 10 nm range. They are formed by the nucleation of individual atoms or molecules, and held together by intermolecular forces such as covalent, ionic or metallic bonds, and Van der Waals forces [79]. Clusters belong to the wider class of nanoscale structures called nanoparticles (NPs), which may have dimensions as large as 100 nm [80]. In this thesis, the term NP is used to describe particles larger than 10 nm.

The atoms at the surface of any object often have very different properties than those that make up the bulk. Such atoms have fewer neighbours, resulting in a locally asymmetric charge distribution and dangling surface bonds. These dangling bonds increase the energy of the surface, changing the way that the surface interacts with its surroundings. Due to their high surface to volume ratio, nanoclusters have a much greater proportion of surface atoms than macroscopic objects, meaning that the effects of surface atoms may be dominant in some circumstances [81]. Also, the tiny size of nanoclusters makes them subject to quantum mechanical effects [82]. These factors result in uniquely size-dependent properties such as: melting point, chemical reactivity, optical and electrical properties [83-85].

It has been proposed that the unique and intriguing properties of clusters/NPs could be exploited in the creation of new materials and devices. Although the ability to observe and manipulate matter on such small scales has been a relatively recent accomplishment of modern science, already many applications of NPs have been discovered, such as in sun-screens and cosmetics, solar and fuel cells, and in medical treatments [85-87].

In this thesis, metal (Sn) clusters (~7 nm) are produced using the inert gas aggregation (IGA) technique [88,89]. IGA introduces vaporised or sputtered atoms into a flow of low-temperature inert gas where they aggregate into clusters. Cluster size can be controlled by the flow, temperature and pressure of the inert gas and the vaporisation/sputtering rate. Clusters produced using IGA form a beam which is incident upon a prepared substrate (Si/Si_3N_4) forming a cluster film. As the density of the film increases, the clusters coalesce into larger NPs (~50 mn). The substrate features two 300 μm wide planar electrodes separated by a 100 μm gap. A 2D percolating NP network is formed when a certain fraction of the area bound by the electrodes is filled with NPs. Detailed discussion of the PASN fabrication procedures can be found in Chapter 2.

1.4.2 Coalescence

When nanoclusters are bought into contact with each other, an agglomerative process called coalescence occurs [90,91]. In much the same way that two water droplets merge together, dangling bonds on the surface of each cluster are free to bind which reduces the number of surface atoms, and equivalently the total surface energy. When enough clusters are present, coalescence will continue to occur until the decrease in surface energy is less than the energy required to rearrange the clusters. If clusters have been exposed to reactive gas species (such

as oxygen or sulphur) then the surface atoms can bind to gas ions making them much less susceptible to coalescence [92].

In this thesis, when clusters land on the substrate they coalesce resulting in larger structures; nanoclusters of diameter ~7 nm can coalesce to form NPs which have dimensions of 20 – 50 nm [93]. The deposition environment contains a small partial pressure of moist air which aids partial oxidation of the cluster surfaces and is used to control the level of coalescence. This procedure has been shown to stabilize the resulting devices which can then operate for many weeks [16].

1.4.3 Percolating Networks

1.4.3.1 Percolation theory

Percolation theory, as first proposed by Broadbent and Hammersley in 1957 [94], describes the way liquid traverses through a randomly distributed solid media. As the density of the solid medium increases, the probability of a pathway which is traversable by the liquid decreases. Since then, many different percolation models have been developed [95]. Of particular relevance to this thesis are theories of 2D percolation and the extension of these theories to explain the conductance of random resistor networks [96].

Figure 1.10 gives a schematic summary of several different 2D percolation models [95]. Figure 1.10 (a) shows *bond percolation*, where the nodes of a 2D $N \times N$ lattice are randomly connected with some probability p. In the limit $N \to \infty$, there exists a continuous pathway between opposite edges (yellow) when the probability p exceeds the critical probability $p_c \approx 0.50$. This critical probability is called the *percolation threshold* and is the critical point in a second-order phase transition between the unconnected and connected phases of the lattice. p is then the *order parameter* which is dominant in determining the connectivity of the system regardless of exactly which nodes are connected by bonds.

Figure 1.10 (b) shows *directed bond percolation*, a variant of bond percolation whereby each bond can only transmit in one direction. Figure 1.10 (c) shows *site percolation* where the nodes of a lattice are randomly occupied with probability p. Adjacently occupied sites are considered to be connected and the same second order phase transition occurs at the critical probability p_c. Figure 1.10 (d) illustrates *continuum percolation*, where the lattice is replaced with an $Nd \times Nd$ area. Discs of diameter d are then placed randomly within the area and allowed to overlap. The order parameter p in this case is the fraction of the surface

area which is covered by discs. As $N \to \infty$, there exists a continuous pathway between opposite edges for a critical surface coverage $p_c \approx 0.68$. It has been shown [17] that this model of overlapping discs provides a good description of the conductance of networks of deposited nanoparticles. Figure 1.10 (e) and (f) show *continuum percolation with tunnelling* for $p < p_c$ and $p \sim p_c$ respectively. The red symbols indicate small gaps between groups of well-connected particles through which tunnelling currents might flow. For $p < p_c$,

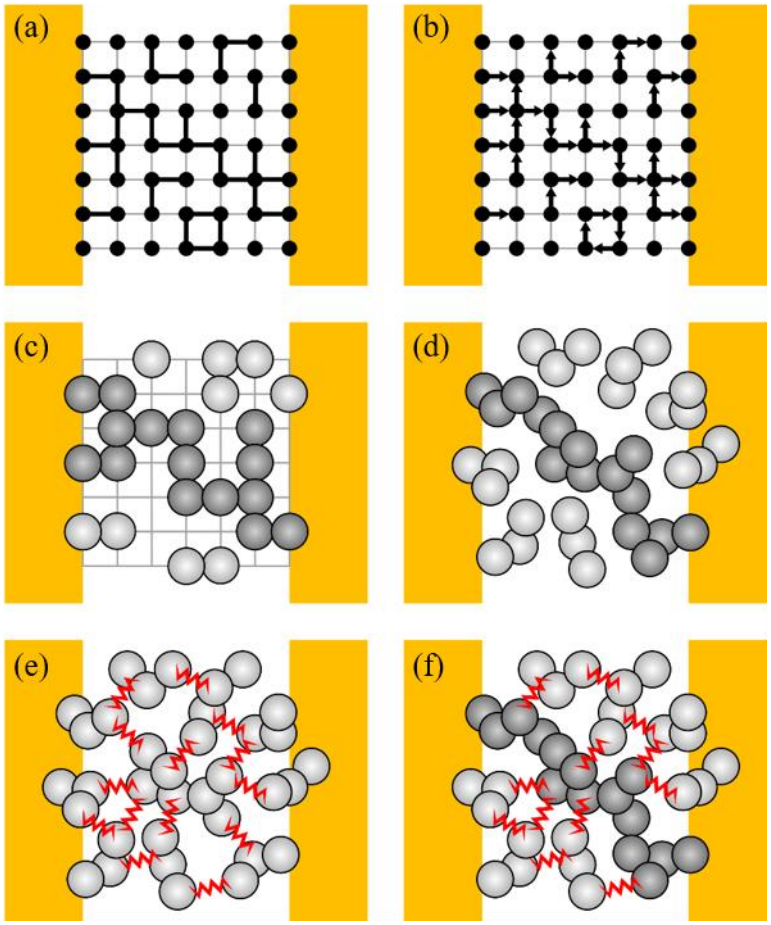


Figure 1.10: Illustration of different 2D percolation models. (a) *Bond Percolation*. Bonds connect nodes with probability p. Note that a pathway exists between the contacts (yellow) for $p > p_c \approx 0.5$ (in the limit of large system sizes). (b) *Directed Bond Percolation*. Bonds are directional, allowing current flow in only one direction. (c) *Site Percolation*. Nodes in the network are occupied with probability p. (d) *Continuum Percolation*. Discs are randomly placed between the contacts and are allowed to interpenetrate. For large system sizes a continuous pathway exists between the contacts for $p > p_c \approx 0.68$. (e) *Continuum percolation with tunnelling at* $p < p_c$. In the continuum model, and near to but below the percolation threshold, quantum tunnelling between particles is possible for small gap sizes (red symbols), allowing conduction even when no continuous pathway exists. (f) *Continuum percolation with tunnelling at* $p \sim p_c$. Tunnelling connections coexist with a continuous pathway between the contacts. [97]

transmission can only occur via tunnelling, but for $p \sim p_c$ there exists a complete connection across the network which is in parallel with incomplete pathways that contain tunnel gaps.

1.4.3.2 Percolating Atomic Switch Networks

The PASNs studied in this thesis are modelled as systems of continuum percolation with tunnelling [17]. The discs represent metal NPs formed by the coalescence of randomly deposited clusters, and the network boundaries represent two pre-fabricated gold (Au) electrodes which sit on a silicon nitride (Si_3N_4) surface. As metal clusters land randomly between the electrodes, the initial probability of clusters overlapping is zero. The proportion of the surface between the electrodes which is covered by clusters (p) grows as the deposition of clusters proceeds. As p increases, so does the probability of clusters overlapping, where they coalesce to form small connected regions on the surface. At the critical surface coverage p_c (i.e. the percolation threshold) a phase transition occurs: a connected region will span the entire gap between the two electrodes.

Shortly before the percolation threshold is reached, many incomplete pathways exist between the electrodes which are disrupted by small gaps. At this surface coverage ($p < p_c$), the only available conduction mechanism is electron tunnelling [98] through the small gaps. Therefore, the conductance (G) depends exponentially on p, since for higher surface coverage, the average size of tunnel gaps is reduced [18]. At this coverage, there are no completed pathways, and the I-V characteristic across the network is non-linear [17].

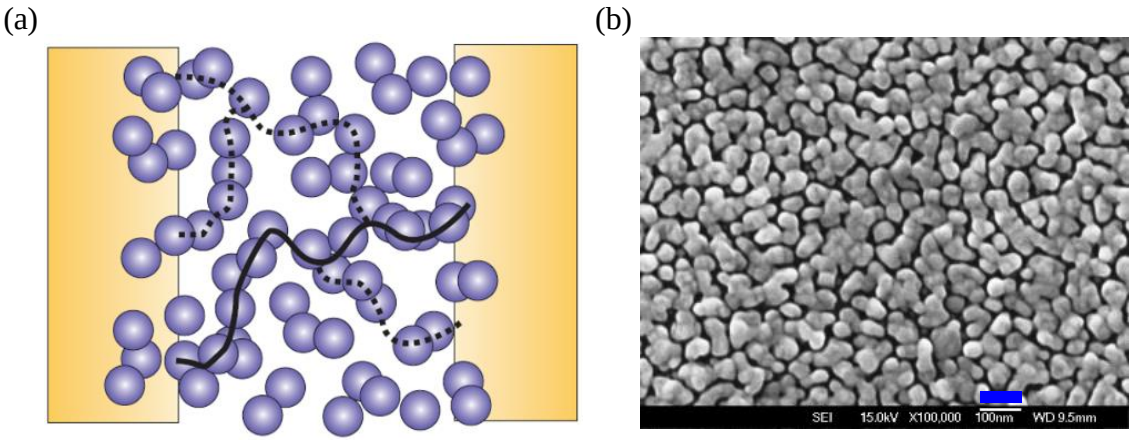


Figure 1.11: Percolating Atomic Switch Networks. (a) Schematic diagram of a PASN. The solid black line represents a metallic connection while the dotted line shows incomplete pathways which are interrupted by tunnel gaps. (b) An SEM image of a zoomed region (scale bar = 100 nm) of a PASN showing the coalesced NP structure which contains many gaps. Reproduced with permission from Ref. [19].

Conversely, if clusters are deposited beyond the percolation threshold ($p > p_c$), there exist several completely metallic pathways (i.e. tunnel gaps have been filled by incoming clusters), and the current flow is then dominated by purely Ohmic connections between the electrodes. At this coverage, the network conductance follows the power law: $G \propto (p - p_c)^t$, where for two-dimensional systems t has the universal value 1.3 [99].

The PASNs studied in this thesis were deposited close to, but slightly above the percolation threshold ($p \gtrsim p_c$). Figure 1.11 (a) shows a schematic diagram of a PASN and Figure 1.11 (b) shows a scanning electron micrograph of the coalesced NPs. At this coverage, the current flow through the network is primarily facilitated by completely metallic connections (solid line in Figure 1.11 (a)), but there also exist many incomplete pathways (dashed lines) which are interrupted by tunnel gaps. When a voltage bias is applied across the two electrodes, huge electric fields exist within the tunnel gaps which lead to additional tunnelling currents (Section 1.4.4). If the bias is sufficiently large (i.e. above some voltage threshold), then atomic switching processes can occur within the tunnel gaps (Section 1.4.5).

1.4.4 Tunnelling Regime

When the voltage V applied across the two electrodes of a PASN is below a threshold voltage V_{thres}, the PASN is said to be in the *tunnelling regime*. There is no switching activity in the tunnelling regime because such processes are electric field driven, and below V_{thres}, the electric field is insufficient. The network topography (i.e. the lay-out of current pathways) is therefore static. Current flow I is facilitated by both complete metallic pathways (Ohmic conduction) and by transmission across small gaps in incomplete pathways (tunnelling conduction).

Ohmic conduction is described by Ohm's law which states that $G = I/V$. Ohms law describes the conduction of an ideal resistor for which the I-V relation is linear, as shown by the red curve in Figure 1.12 (a). The conductance is proportional to the slope of the I-V curve, which is the same for all voltages, as shown in Figure 1.12 (b).

Tunnelling conduction [98] is a quantum mechanical phenomenon whereby electrons can penetrate a potential barrier which is forbidden by the laws of classical mechanics. The details of quantum tunnelling can be found in Ref. [100]. Tunnelling conduction does not follow Ohm's law: there is a disproportionate increase in I relative to V which leads to the nonlinear

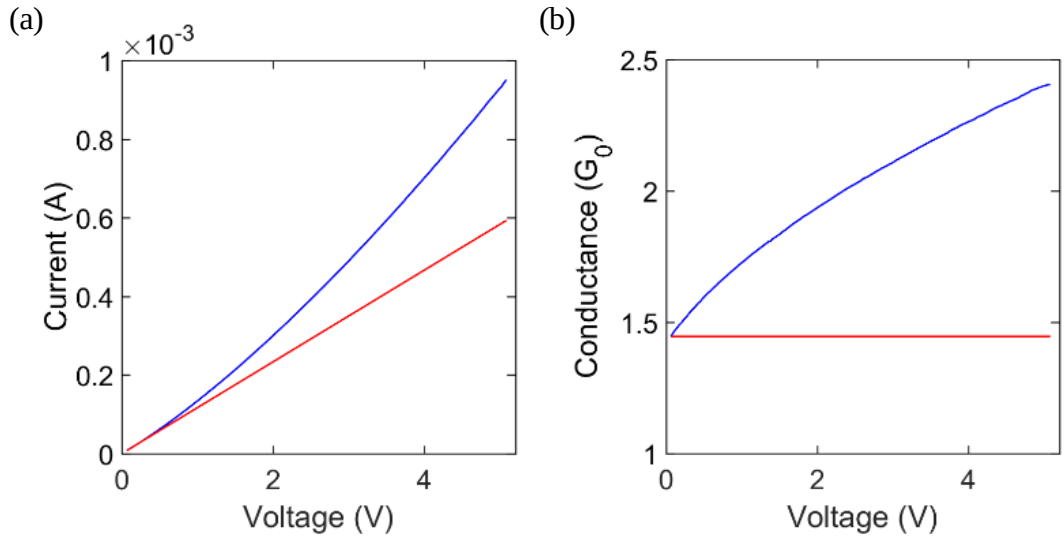


Figure 1.12: Ohmic vs non-ohmic conduction. (a) Current-voltage characteristic ($I(V)$) and (b) conductance-voltage characteristic ($G(V)$) for an ideal resistor (red) and a PASN in the tunnelling regime (blue). The ideal resistor shows linearity (i.e. Ohm's law), while the PASN is nonlinear due to tunnelling conduction.

I-V characteristic shown (blue) in Figure 1.12 (a) and the increasing conductance with voltage shown (blue) in Figure 1.12 (b).

There are several models of tunnelling conduction in nanoscale electronic systems, including the Kaiser model and the Simmons model. The Kaiser model [101] is a generic, phenomenological model for tunnelling conduction in networks which has been shown to fit various nanoscale electronic systems. It states:

$$G = \frac{g_0(T)\exp\left(V/V_0\right)}{1 + h[\exp(V/V_0) - 1]} \tag{1}$$

where $g_0(T)$ is the zero-bias conductance, and h is the ratio of $g_0(T)$ with the saturation conductance. The parameter V_0 is closely related to the barrier height.

The Simmons model [102] is a theoretical model of tunnelling between two metal electrodes through a thin insulating film. It states that the current density for a generalized tunnel barrier is:

$$J = J_0\left(\varphi_B\, e^{-A\sqrt{\varphi_B}} - \left(\varphi_B + eV\right) e^{-A\sqrt{\varphi_{B-eV}}}\right) \tag{2}$$

with

$$A = \frac{2\beta s_B}{\hbar}\sqrt{2m} \tag{3}$$

and

$$J_0 = \frac{e}{4\pi^2\beta^2\hbar {s_B}^2} \quad (4)$$

where φ_B and s_B represent the barrier height and width respectively and the symbol β is a dimensionless correction factor of order unity.

Unlike the Kaiser expression, the Simmons expression contains physically meaningful parameters, such as the average barrier height and width. This is useful when modelling a single junction as in Ref. [16]. However, when considering the conductance of an entire PASN which contains many tunnel junctions and Ohmic pathways, the Kaiser model is just as useful. In Section 5.2, the Kaiser model is fitted to the G-V characteristic of a PASN and the fitted parameters used to model the behaviour of PASNs in the tunnelling regime.

1.4.5 Switching regime

When the voltage V applied across the two electrodes of a PASN is above the threshold voltage V_{thres}, the PASN is said to be in the *switching regime*. For $V > V_{thres}$ there exists sufficiently large electric fields within tunnel gaps to cause the formation of atomic scale filaments. Conversely, formed filaments are subject to sufficiently high current densities to cause filament rupture. Tunnel gaps which exhibit this behaviour are called *atomic switches*.

1.4.5.1 Atomic Switches

There is considerable evidence that atomic scale filaments can form in nanoscale gaps between NPs [18,19,103]. *Electric field induced surface diffusion* (EFISD) is a process whereby atoms on the surface of a solid become mobile under the influence of a strong electric field ($\vec{E} > 1$ V/m) [104]. When $V > V_{thres}$, the electric field strength within some of the tunnel gaps in a PASN is sufficient to enable diffusion of atoms on the NP surface which then drift into the gap forming a protrusion. This protrusion causes the gap to narrow, further increasing the strength of the electric field.

For electric fields greater than 25 V/m, a jump to contact may occur via *electric field induced evaporation* (EFIE) [105]. In this case, atoms at the narrowest point of the gap are ionized and ejected from the metallic surface, forming a protrusion from the opposite side of the gap. When the two protrusions meet via a single atomic contact, the filament is formed and a quantized conduction pathway exists between the NPs [19]. The quantum of conductance is $G_0 = 2e^2/h = 7.75\times10^{-5}$ Siemens. Figure 1.13 contains schematic diagrams

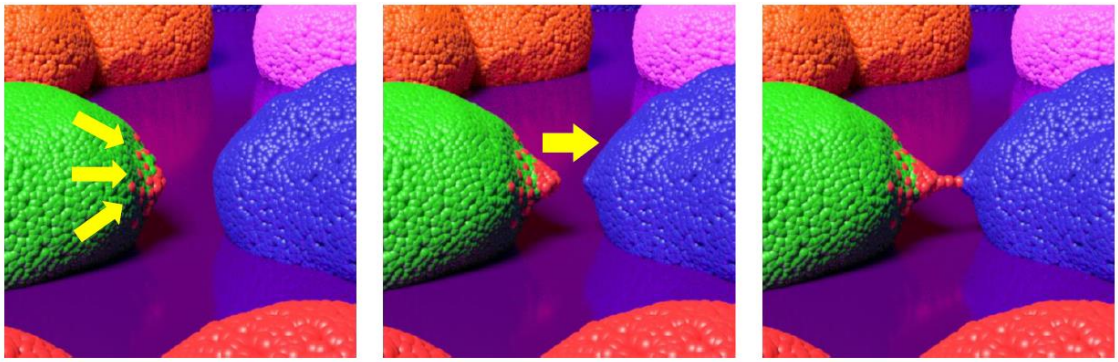


Figure 1.13: Schematic of atomic filament formation. (Left) Atoms on the surface of a NP undergo EFISD to form a protrusion in the gap. (Middle) The protrusion causes the gap to narrow, further increasing the electric field. (Right) EFIE then causes a jump to contact forming an atomic scale filament that bridges the gap between NPs.

of the EFISD and EFIE processes, and the resulting atomic scale filament. This process results in a stepwise increase in the conductance of the network.

The formation of an atomic scale filament can connect previously disjointed sections of the network, creating an Ohmic conduction pathway[1] which spans the network. When this occurs, the filament is subject to very high current density, and may be subject to rupture by *electromigration*. The atoms within a filament experience a force induced by momentum transfer from collisions with conduction electrons. This causes the atoms to migrate in the direction of electron flow, and when current densities exceed $\sim 2\times10^{8}$ A/m^{2}, the resulting force on the atoms is great enough to cause mechanical breakdown of the filament [107]. The observable result of this process is a stepwise decrease in the conductance of the network.

1.4.5.2 Switching Events

Figure 1.14 shows an example of a time-resolved measurement of G while applying a 6 V DC bias. The stepwise increases and decreases in G correspond to filament formation and filament destruction respectively. The occurrence of either the formation or destruction of a filament is called a *switching event* and the size of each switching event is defined as the total change in G (called ΔG) that results from the switching event. Figure 1.14 shows switching events of many different sizes occurring stochastically in time: typical behaviour of a PASN in the switching regime.

[1] Current through an atomic scale filament occurs via ballistic transport [106]. Although the conduction mechanism in the filaments is different than in the NPs (where there is resistivity due to scattering), both obey Ohm's law and so the completed pathways are Ohmic.

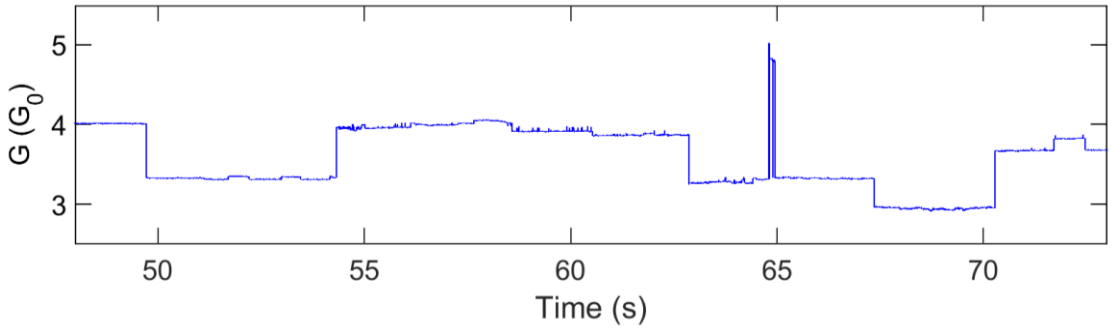


Figure 1.14: Example of PASN conductance switching. A time resolved measurement of the conductance G of a PASN under a 6 V DC bias. Increases in G correspond to filament formation via EFISD/EFIE and decreases in G correspond to filament destruction via electromigration; both are termed *switching events*. Although the conductance *of the filaments* is quantized, their contribution to the conductance *of the network* depends on where the switching junction is located in the network. Thus, the size (ΔG) of a switching event can range over several orders of magnitude.

Each time a switching event occurs, the network topology (i.e. the layout of conduction pathways) changes. Consequently, there is a redistribution of electric fields and current densities throughout the network which may induce further switching events at other switching junctions. In this case, switching events are said to be *correlated* because each event happens as a result of some previous event(s). Switching activity may therefore occur in bursts or "avalanches" which are an indicator of criticality and are discussed extensively in Chapter 3.

Finally, it has been demonstrated [15] that the average number of switching events per unit time is dependent on the applied voltage. This is because larger biases induce stronger electric fields within the tunnel gaps in the network resulting in the activation of a larger number of atomic switches. Concurrently, larger currents flow through conduction pathways in the network leading to increased current densities in atomic filaments. This leads to an increased number of filament ruptures. In addition to the increase in the number of active switching sites, the EFISD/EFIE and electromigration processes responsible for switching activity are accelerated by larger electric fields and current densities. Thus, applying larger bias voltages increases both the number of active switches, and the rate at which filaments form and rupture. The voltage dependence of the switching rate is used to model the switching regime nonlinear dynamical node (SRNDN) in Section 5.3.

1.4.6 Dynamic Switching Threshold

The voltage threshold V_{thres} which separates the tunnelling and switching regimes of PASN behaviour is dynamic. V_{thres} is initially ~3 V, but changes depending on the bias voltage

applied to the PASN. Larger bias voltages (e.g. $>$ 8 V) applied for extended periods of time cause the network to "fatigue", whereby some switching junctions become permanently unable to switch. This results in an increase in V_{thres} meaning that larger bias voltages are required to induce the same level of switching activity.

The exact processes that cause fatigue are not well-understood, nor are they a focus of this thesis. Fatigue does however enable larger bias voltages to be applied while remaining in the tunnelling regime. This property is exploited in Section 6.1 where a PASN was intentionally fatigued by applying 10 V DC for many hours in order to then apply bias voltages $\leq$ 8 V without inducing unwanted switching events.

1.5 Summary and Thesis Layout

Chapter 1 has introduced the brain as a biological computer which can perform recognition and prediction tasks with paramount efficiency, as compared with conventional CMOS/von Neumann computing architectures. ANNs were then described as software-based approaches to brain-inspired computing, which mimic aspects of information processing in the brain. However, ANNs rely on conventional computing hardware which is fundamentally different than the architecture of the brain and thus ANNs consume many orders of magnitude more energy than the brain. Neuromorphic computing was proposed as a potential solution to this problem. Radically new and innovative computing architectures are required that are naturally suited to performing brain-like information processing. Realization of such architectures are projected to allow neural computing with the efficiency of the biological brain.

This thesis explores PASNs as candidates for novel neuromorphic computing architectures. PASNs exhibit complex nonlinear dynamics in response to electrical signals making them suitable to act as physical reservoirs in a RC scheme. The layout for the remainder of this thesis is as follows.

Chapter 2 details the materials and experimental procedures used to fabricate PASNs, as well as the analytical procedures used to analyse experimental data.

Chapter 3 reviews several emergent properties of biological neuronal networks, including critical avalanche dynamics, and demonstrates that PASNs share these same properties. It is definitively shown that PASNs are critical systems, a feature which other neuromorphic approaches lack.

Chapter 4 then introduces a computational framework called time-delay reservoir computing (TDRC). A review of relevant literature is presented, and some common tasks used to benchmark novel neuromorphic architectures are described.

In Chapter 5, several numerical implementations of TDRC are used to perform classification and prediction tasks. One of these implementations utilizes a model of a PASN in the tunnelling regime, while another uses a model of a PASN in the switching regime.

Finally, experimental attempts at TDRC using real PASN devices in the tunnelling regime are detailed in Chapter 6.

Chapter 2

Materials and Methods

This chapter describes the materials and the experimental and analytical procedures used in this thesis. Section 2.1 gives an overview of the materials used to fabricate PASN devices: nanoparticles made from tin (Sn) and the silicon/silicon nitride (Si/Si_3N_4) substrate. Section 2.2 outlines the procedures used to fabricate PASNs and to perform subsequent electrical measurements. Substrate preparation and the cluster deposition system are described in detail. Finally, Section 2.3 discusses the analytical procedures used to characterise the electrical properties of PASN devices. This includes identifying switching events from time-resolved conductance measurements, statistical analysis of those switching events, and fitting models to their statistical distributions from which parameters are extracted.

2.1 Materials

2.1.1 Silicon/Silicon Nitride

Si wafers with a 200 nm Si_3N_4 passivation layer were used as substrates for the percolating nanoparticle networks. Many properties of Si_3N_4 make it an excellent substrate for microelectronics, such as its hardness, low chemical reactivity, and electrical impermeability [108]. Essential to the fabrication of stable PASN devices are the atomic scale smoothness of the Si_3N_4 layer which allows controlled coalescence of clusters on the surface (Section 1.4.2), and its electrical insulating properties which restrict current flow to the nanoparticle network rather than through the Si wafer itself. Of particular relevance to the neuromorphic theme of this thesis is the compatibility of these substrates with existing CMOS technology; a beneficial attribute when it comes to upscaling and integrating successful architectures.

2.1.2 Tin

The nanoparticles (NPs) which comprise the percolating network of a PASN are made from tin (Sn). Sn is a post-transition metal which is situated in group 14 of the periodic table. It has a magic atomic number of 50, meaning that that all of its nuclear shells are filled, giving rise to unique properties [109]. For example, it has 10 stable isotopes, more than any other chemical element, and has 2 common allotropes at room temperature. The most stable of the two is β-Sn, a malleable and ductile metallic form which is silver in colour and has body-centred tetragonal structure. At temperatures below 286.35 K (13.2 ℃) β-Sn morphs into the diamond cubic structure α-Sn, which in contrast is brittle and grey, and exhibits no metallic behaviour. This transformation proceeds slowly above ~243 K (-30 ℃), but it has been observed that the transformation can be catalysed by the presence of either α-Sn (autocatalysis) or the 4+ oxidation state (SnO_2). The latter is the most stable of two possible oxidation states that exist for tin, the other being 2+ (SnO). Metallic Sn oxidizes naturally in the atmosphere, forming a thin SnO_2 passivation layer which protects the bulk from further oxidation [110]. Ref. [111] shows that the rate for such a reaction, which is initially slow, can be increased under humid conditions. The oxidation of Sn NPs exhibits a strong size dependence [112] by which the self-limiting oxidation which results in the passivation layer

of bulk Sn proceeds to different depths depending on the surface curvature. Furthermore, for NPs smaller than ~ 20 nm, the oxidation may continue until the Sn has been entirely consumed, leaving behind a hollow oxide shell [112]. Thus, the partial pressure of oxygen and humidity inside the deposition chamber (Section 2.2.1.2) must be precisely controlled to allow the Sn NP surfaces to be partially oxidized without proceeding to complete oxidation of the NPs. The resulting NP networks therefore retain their metallic properties, but the presence of SnO/SnO_2 helps to stabilize the networks [16].

2.2 Experimental Procedures

2.2.1 PASN Fabrication

PASNs are fabricated by depositing metal nanoclusters onto a Si_3N_4 surface between two Au electrodes. This Section details the preparation of the Si_3N_4 substrate and the subsequent NP deposition. The procedures outlined here are also discussed extensively in Refs. [93,113].

2.2.1.1 Substrate Preparation

This section details the preparation of the passivated silicon wafers which are used as the substrates for PASN fabrication. In order to test the electrical properties of the NP networks after deposition, the first treatment of these wafers is the deposition of two 50 nm thick Au/NiCr (gold-nichrome) electrodes. This is achieved by shadow mask evaporation within the ultra-high vacuum (UHV) chamber of an Edwards Auto 306 thermal evaporator. A thin metallic sheet with a 5 x 5 array of H-shaped perforations (the shadow mask) is suspended below the nitridized face of the Si wafer. Further below, a small tungsten crucible containing a raw Nichrome (NiCr, 99.5%) is subjected to an electrical current, the resistance of which heats the crucible, causing the NiCr to slowly evaporate. The perforated sheet acts as a stencil, allowing the evaporated NiCr atoms to pass through the perforations, forming 5nm thick H-shaped layers of NiCr on the substrate. A 100 µm thick wire atop the shadow mask runs down the centre of each column of perforations, vertically dissecting the H-shaped NiCr layers, as shown in Figure 2.2 (a). The result is a Si wafer which has 25 pairs of 300 µm wide electrodes, each of which features a 100 µm gap. Following the NiCr deposition, the exact same evaporation procedure is used to deposit a 45 nm thick layer of Au (99.99%). The nichrome forms a wetting layer which aids in the adhesion of the Au contacts to the Si_3N_4

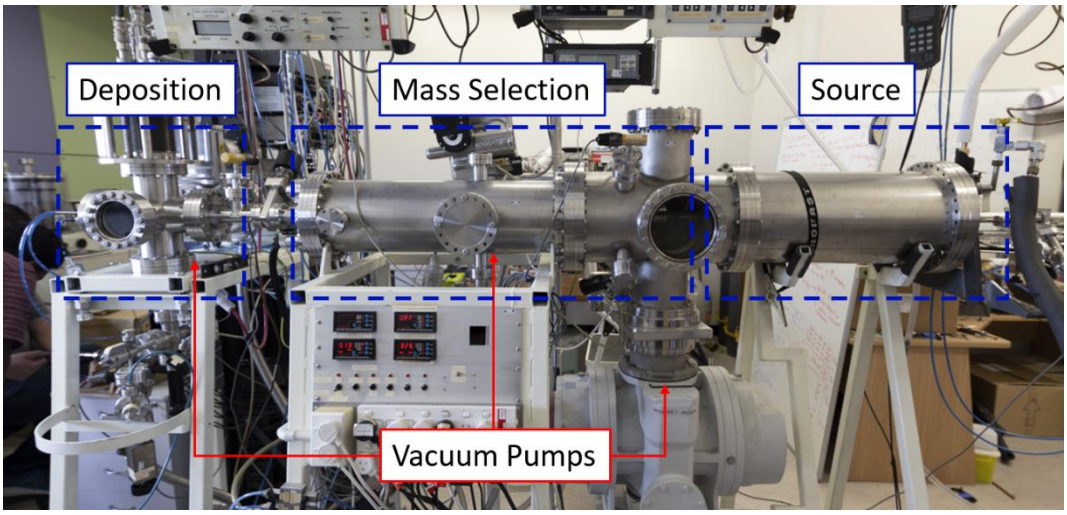


Figure 2.1: Photograph of the deposition system. Clusters produced at the source are propelled towards the substrate in the deposition chamber due to the pressure gradient created by the three stage pumping system.

surface. The wafer is then dissected into 25 10 x 10 mm samples, one of which is shown in Figure 2.2 (b).

Once the electrodes are fabricated, the electrode profiles are measured using a Dektak profilometer. This ensures that the gap between the electrodes is of desired size and shape, and that there are no Au/NiCr particles within the gap. The substrates are then cleaned in acetone to remove any contaminants. Because acetone is a fast evaporating solvent, a successive rinse in isopropyl alcohol (IPA) is required to wash away any dissolved contaminants remaining on the surface. Following the IPA rinse, the substrates are blow-

(a) (b)

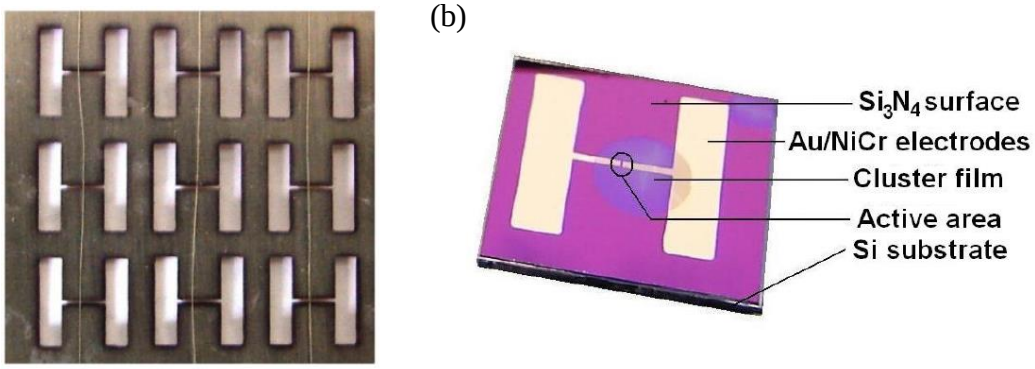


Figure 2.2: Substrate preparation. A shadow mask used in the deposition of the Au/NiCr electrodes onto the Si3N4 substrate. A 100 µm thick wire runs down the centre of each column of perforations providing the gap in which cluster networks can be deposited (b) A photograph of a fabricated sample. The 10 x 10 mm wafer features prefabricated Au/NiCr electrodes, separated by a 100 µm wide gap. The darker region in the centre shows the area which has been deposited with Sn clusters. [93]

dried with 99.99% N_2, and examined using an optical microscope. If no contaminants remain in or near the active area between the electrodes, the substrates are mounted to the cold finger cryostat for cluster deposition.

2.2.1.2 Cluster Deposition

Nanoparticles are deposited onto the pre-prepared substrates using a UHV cluster deposition system [89]. The deposition system (shown in Figure 2.1) is comprised of three primary components: the source chamber, where clusters are produced using a magnetron sputter source; the mass selection chamber where clusters can be selected based on their size; the deposition chamber where clusters make contact with the pre-prepared substrates. A three-stage pumping system creates a pressure gradient along the length of the apparatus which carries nanoclusters produced at the source down to the substrate in the deposition chamber. The components of the deposition system are described extensively in Ref. [113] and so only a brief description is given here.

Pumping System

The vacuum inside the cluster deposition system is maintained by a three stage pumping system which utilizes several different types of pumps.

Rotary pumps [114] are used to create low vacuum (~10^{-3} Torr) to provide backing and fore-pumping to the roots pumps and molecular turbo pumps. Rotary pumps capture gas using mechanical techniques, often using an electric motor which drives a piston or diaphragm. Captured gasses are then expelled through an exhaust outlet. The moving parts are often lubricated with oil and so filters are required in order reduce the chance of contaminating the vacuum environment.

Roots vacuum pumps [114] are used to evacuate large volumes of gas quickly and efficiently. They can provide good quality vacuum up to ~10^{-5} Torr. Roots pumps are positive displacement lobe pumps, in which gas is compressed and compelled out due to two-lobe structures rotating at high speed. The structure and the rotation of the lobes generates low pressure on one side which helps to remove the gases out of the chamber, and high pressure on the other side which pushes the gases to the inlet of a backing rotary pump where they are expelled to the environment.

Turbo molecular pumps [115] are capable of maintaining UHV up to ~10^{-10} Torr. They use momentum transfer to capture gas molecules. This is achieved using a series of spinning

turbines called *rotors* and stationary turbines called *stators*. Gas molecules collide with the angled blades on a rotor, gaining momentum. This causes a pressure increase between the rotor and the stator due to the relative motion between them. Subsequent rotor-stator pairs cause the increased pressure to rise as captured gas makes its way from inlet to the outlet, where it is expelled by a backing pump. The turbines rotate at a large number of revolutions per minute and hence require a cooling system to relieve thermal load on the bearings.

Cluster Source Stage

The source chamber houses a magnetron sputter source [116], a liquid N_2 cooling system and a supply of inert Ar gas. A schematic diagram of the source chamber is shown in Figure 2.3. The magnetron sputter source contains a metal disc called a target[2] which supplies the material for the nanoparticles. The target acts as a cathode under strong DC bias. Ar gas is bled into the source chamber close to the surface of the target where it is ionized by the strong electric field. The Ar^+ ions then undergo rapid acceleration towards the negatively charged target. The bombardment of Ar^+ ions transfers energy to the metal surface atoms, and if this energy transfer exceeds the binding energy[3] of the target material, causes atoms to sputter out of the surface of the target. A permanent magnet situated at the back of the target captures any secondary electrons produced during ionization (or during ionic bombardment of the target) and traps them in helical trajectories close to the target surface. These electrons cause further ionization of the Ar gas and allow plasma creation at low pressure.

Sputtered atoms are carried by the Ar gas towards the nozzle at the end of the source chamber due to the strong pressure differential between the source chamber and the mass-selection chamber. As the metal atoms traverse the distance labelled as the aggregation length in Figure 2.3, they aggregate into clusters. The aggregation length can be adjusted externally, thus controlling the size of the clusters: larger aggregation lengths mean that the time of flight between the target and the nozzle is longer causing increased aggregation and larger clusters. Other factors which determine the distribution of cluster sizes are the bias applied to the target, and the pressures of the source and mass selection chamber, each precisely controlled.

[2] A range of different metals can be used to create the nanoparticles, including Pb and Ag, but the PASNs studied in this thesis consist exclusively of Sn NPs. The target is a disc (50 mm diameter, 6mm thickness) of 99.999% pure β-Sn.

[3] Surface atoms require three times the binding energy to be liberated from the target; approximately equivalent to the heat of sublimation [117]

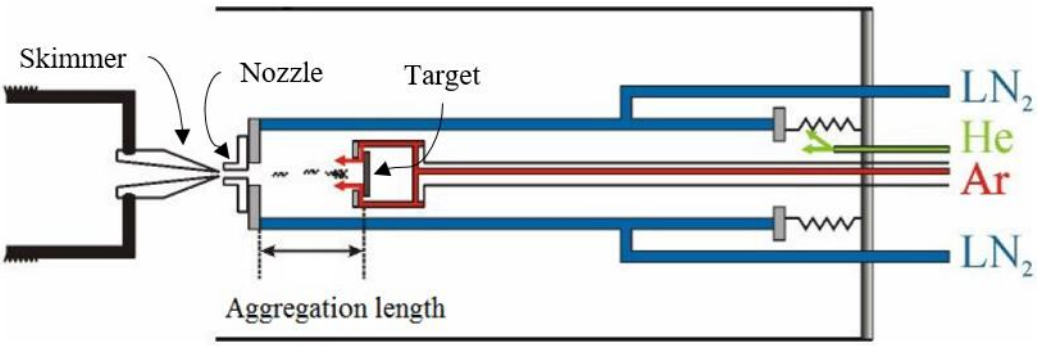


Figure 2.3: Schematic of the source chamber. [118]

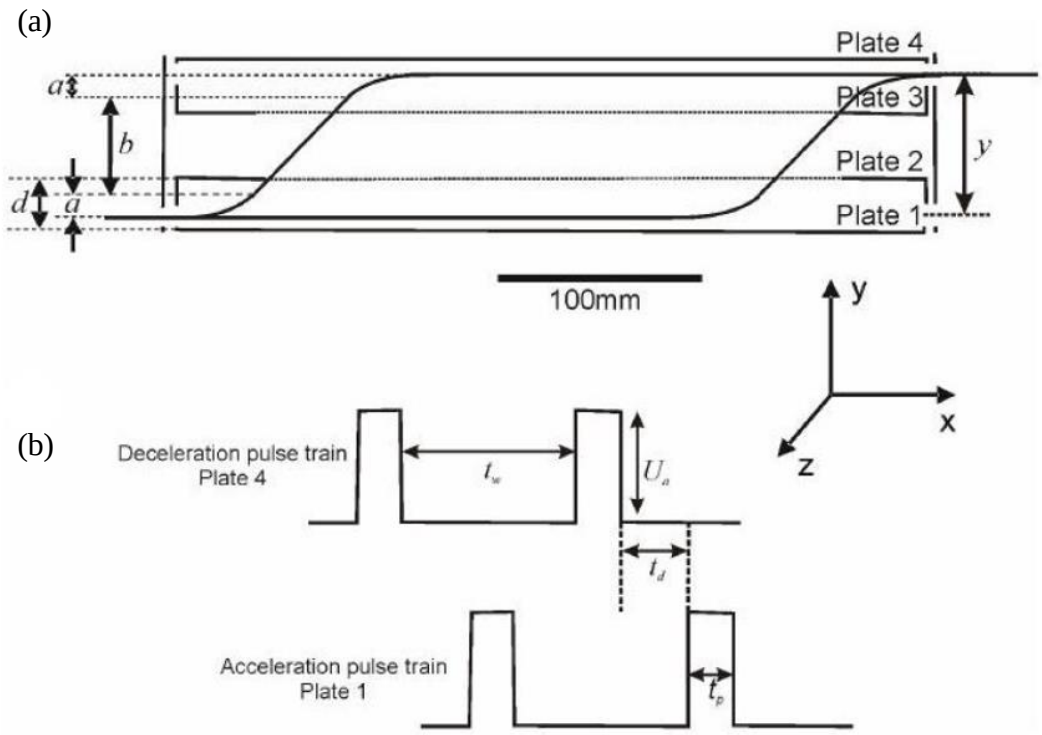


Figure 2.4: Schematic of the Palmer-Issendorff mass selector. (a) Cluster beam trajectory is in the x-direction and aligned with the space between plates 1 and 2. Charged clusters are accelerated in the y-direction by voltage pulses applied across plates 1 and 2 and decelerated by voltage pulses applied across plates 3 and 4. Clusters travel a distance a during the pulses and b in between pulses. (b) Relative pulse timing is used to select a particular cluster mass. Pulses U_p are applied for duration t_p. t_d is the delay time between the acceleration and deceleration pulses. [118]

Liquid N_2 flows through cooling lines which prevent the source from overheating while water lines running to the sputter source prevent the rubber UHV gasket at the back of the target from freezing.

Mass-selection Stage

Clusters produced at the source stage enter the mass-selection chamber through a skimmer [119]. The skimmer is primarily responsible for shaping the plume of clusters (and Ar) which emerges from the source chamber nozzle into a collimated beam. The alignment of the nozzle and skimmer is of utmost importance as a small misalignment can have a drastic impact on the resulting cluster beam.

After exiting the source chamber, the majority of the Ar gas is evacuated from the cluster beam via the first pumping stage which features a roots pump backed by a rotary pump. These pumps maintain the pressure inside the source chamber at ~10^{-5} Torr. The clusters, which have much larger momentum than the Ar atoms, continue on their trajectory towards the deposition chamber. The pressure gradient inside the mass chamber is maintained by a second pumping stage: a molecular turbo pump backed by a rotary pump.

Inside the mass-selection chamber is a Palmer-Issendorff mass selector [120]. The mass selector consists of two pairs of metallic plates which are parallel with the trajectory of the beam, as shown schematically in Figure 2.4. A fraction of clusters within the beam are ionized and are therefore subject to Lorentz force when moving through an electric field. The clusters enter the space between plates 1 and 2, where the ionized clusters are subject to a positive force in the y-direction exerted by an accelerating voltage pulse applied across plates 1 and 2. The clusters then travel through holes in plates 2 and 3 into the space between plates 3 and 4. A decelerating voltage pulse of equal but opposite amplitude is then applied across plates 3 and 4, removing the vertical velocity component gained during the accelerating pulse. The clusters then leave the mass selector in the x-direction through an aperture.

Because the acceleration pulse causes light clusters to gain more vertical velocity than the heavy clusters, they travel farther upwards during the time between pulses. Consequently, for a certain pulse magnitude U_p, pulse duration t_p and delay time t_d, there is a very small range of cluster sizes which are aligned with the exit aperture.

The mass selector can be used to limit the sizes of clusters in the beam to a narrow range, at the expense of reduction in mass flux. Alternatively, it can be used to measure the mass spectrum of the cluster beam by placing a Faraday cup at the exit aperture of the mass

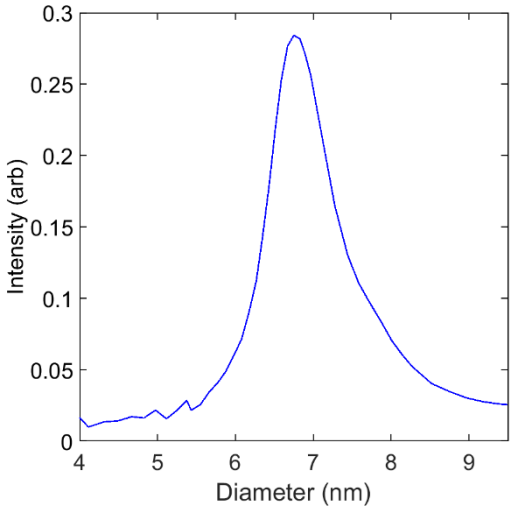


Figure 2.5: Typical mass spectrum of the cluster beam.

selector. A Faraday cup can be used to infer the number of incident clusters by measuring the resulting current induced when the charged clusters make contact with the Faraday cup and become neutral. Mass spectroscopy is performed by iteratively changing the magnitude of the acceleration/deceleration pulses and measuring the relative number of incident clusters for each magnitude using the Faraday cup. In this thesis, the mass selector was used only to conduct mass spectroscopy prior to the deposition of each PASN device. A typical mass spectrum of the cluster beam is shown in Figure 2.5. It shows that cluster sizes are distributed as a narrow peak centred at ∼7nm.

Deposition Stage

The deposition chamber (shown schematically in Figure 2.6 (a)) contains a Janis ST-400 cold finger cryostat (shown in Figure 2.6 (b)) on which pre-prepared substrates are mounted. The cold finger cryostat is secured to a linear translator which allows its vertical position to be adjusted externally. The sample holder at the bottom of the cold finger features three sample stages complete with electrical probes for conducting in situ electrical measurements. Two thermocouples and a small heating element are mounted on the sample holder for temperature-dependence measurements.

The atmosphere inside the deposition chamber is at the lowest point along the pressure gradient inside the cluster deposition system. The vacuum is maintained by the third pumping stage: a second molecular turbo pump backed by another rotary pump. During cluster deposition, a small partial pressure of moist synthetic air is introduced to the deposition chamber using a needle valve which is connected to a bubbler containing distilled water. The moisture and oxygen allow the cluster surfaces to partially oxidise which limits the

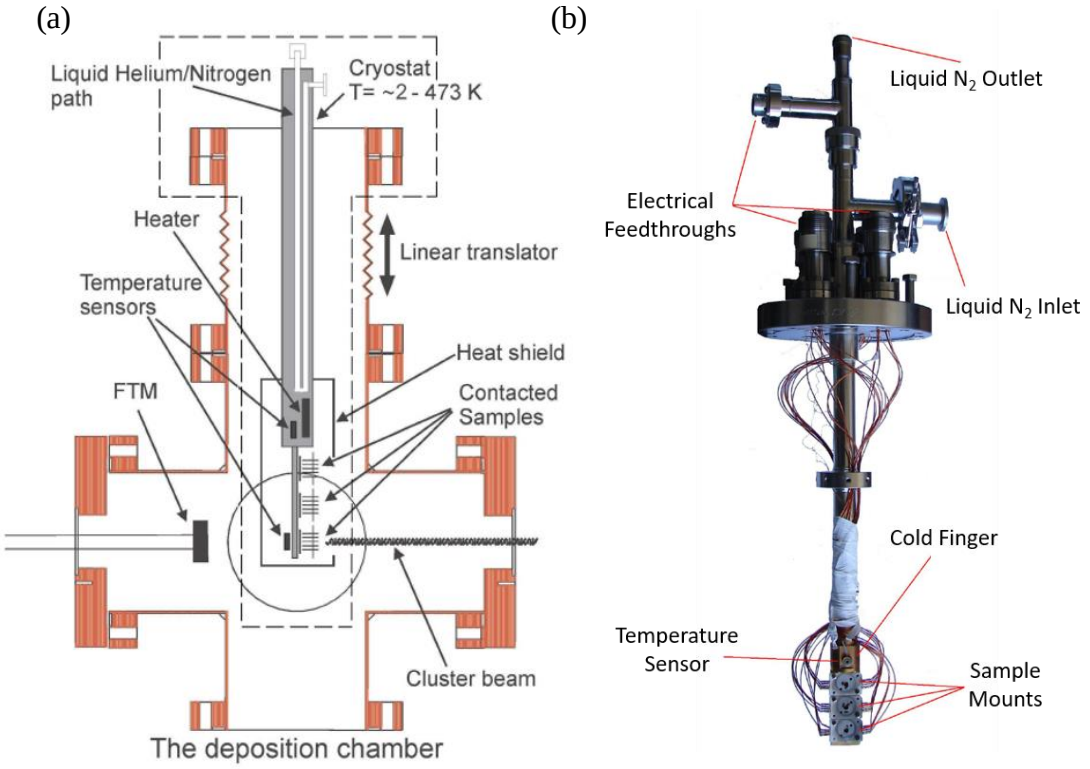


Figure 2.6: Deposition chamber and sample holder. (a) Schematic of the deposition chamber. [118] (b) A photograph of the sample holder [93]

coalescence of the clusters (See Section 1.4.2) after landing on the substrate, leading to stable PASN devices [16].

At the back of the deposition chamber is a Sycon Instruments STM-100/FM film-thickness monitor (FTM) which is used to monitor the deposition rate. Typical deposition rates used to fabricate PASNs are ~0.2 Å/s. Because the sample holder intercepts the cluster beam during deposition (see Figure 2.6 (a)), the rate cannot be measured simultaneously and so it measured both prior to and immediately following the deposition of each of the three devices. The deposition rate can be controlled by adjusting the source bias.

An electronically-triggered, gas-powered interlock separates the mass-selection and deposition chambers and is used to terminate the cluster beam when not depositing clusters and to isolate the deposition chamber once deposition is complete. Cluster deposition proceeds by lowering the externally adjustable cryostat so that one of the three mounted substrates is aligned with the centre of the cluster beam. The interlock is opened, and clusters are incident on the substrate, slowly covering the active surface area between the two electrodes. During deposition, the resistance of the sample is monitored by applying 100 mV across the two electrodes on the substrate and measuring the resulting current. The measured

current remains at zero[4] until the onset of conduction which occurs when the surface coverage between the two electrodes reaches the percolation threshold (see Section 1.4.3). Cluster deposition is then allowed to proceed until the measured current corresponds to a device resistance of 2 kΩ, at which point the beam is terminated by closing the interlock. The cryostat is retracted from the cluster beam and the deposition rate remeasured before lowering the second substrate into the cluster beam and the process is repeated for the remaining two samples. Following the deposition of all three samples, the interlock is closed and the fabricated PASN devices remain in vacuum inside the deposition chamber where they are subject to electrical characterisation.

2.2.2 Electrical Measurements

Conductance measurements of PASN devices were performed by applying a voltage V across the two electrodes of the PASN and taking time-resolved measurements of both V and the resulting current I. Conductance G was then calculated in post-processing using $G = I/V$. Three different measurement systems were utilized to perform conductance measurements in this thesis. For all measurement systems, the measurement circuit was connected using coaxial cables of 50 Ω impedance terminated with BNC[5] connections.

Keithley System

The first system was used to perform low bandwidth conductance measurements. It consists of: an IOtech digital-to-analog converter (DAC) which was used to apply voltages across the PASN electrodes; a Keithley 2000 multimeter used to measure the applied voltage; a Keithley 6514 electrometer used to measure the current flowing through the PASN. The instruments are controlled by a LabVIEW graphical user interface (GUI) via GPIB[6] cables. The Keithley system was limited to a sampling frequency of 5 Hz due the latency between the two independent measurement instruments. However, its low bandwidth allows for averaging (over a 20 ms window) in real time which is desirable for noise reduction purposes. The Keithley system was used to perform the 5 Hz measurements discussed in Section 3.2.3.

[4] The measured current is actually a few pA which is the Keithley 6514 leakage current, but this represents no measurable current flow between the PASN electrodes.

[5] Bayonet Neill–Concelman

[6] General Purpose Interface Bus

Oscilloscope System

The second measurement system utilizes a two-channel digital phosphor oscilloscope (Tektronix TDS5032B), a Stanford SR570 operational amplifier (op-amp) and a TTI Instruments TG-2000 function generator. The TG-2000 was used to apply voltages across the PASN electrodes which were measured directly using one of the oscilloscope channels. The other oscilloscope channel was used in conjunction with the op-amp to indirectly measure the current. Op-amps produce a voltage which is a linear function of the current flowing into them. Thus, by replacing the ground in the measurement circuit with the virtual ground of the op-amp, the current flowing through the circuit can be inferred by measuring the op-amp output voltage with the oscilloscope. The current values were calculated in post-processing by multiplying the measured op-amp voltages by the op-amp gain which was set to 1mA/V. Using a single oscilloscope to simultaneously measure V and I eliminates all of the latency issues associated with the Keithley system. Additionally, the oscilloscope is capable of high-bandwidth measurements up to 500 MHz and features an averaging filter which was used to minimise sampling noise. However, it has a limited memory of 2×10^6 points per recording which limits the total duration of each experiment depending on the sampling rate. The oscilloscope system was used to perform the 5 kHz measurements discussed in Section 3.2.3.

National Instruments (NI) System

The third measurement system comprises a National Instruments PXIe-6378 ADC[7]/DAC and the same Stanford SR570 op-amp used in the oscilloscope system. The PXIe-6378 features 4 dedicated DAC channels and 16 dedicated ADC channels which can simultaneously measure voltages at up to 3.5 MHz. The instrument is controlled using a custom built LabVIEW GUI. Like the oscilloscope system, the op-amp is used to infer the current by measuring its output voltage with one of the ADC channels. Simultaneously, the voltage is measured using another ADC channel. The NI system does not feature a built-in averaging filter to reduce noise and so an averaging filter is applied in post-processing. This involves measuring at a higher sampling rate and then down-sampling by taking the average of multiple recorded points. For example, to obtain a 5 kHz measurement with averaging would mean sampling at 100 kHz and then taking the average of every 20 points so that the final

[7] Analog-to-digital converter

sampling interval after averaging is 200 μs. This procedure has a very similar effect to using the built-in averaging filter of the oscilloscope system.

There are two[8] (relevant) advantages of using this measurement system over the other two systems. Firstly, it is capable of streaming raw measured data directly to a solid-state hard-drive (SSD) which eliminates the experiment duration constraint imposed by the limited memory of the oscilloscope system. Secondly, the DAC channels can each be programmed to output arbitrary waveforms. This is essential for time-delay reservoir computing (Chapter 4) where sequences of masked input values must be applied to the PASNs as voltages. Thus, the NI system was used to perform all of the measurements (5 kHz after averaging) presented in Chapter 6.

2.3 Analytical Procedures

This Section describes the analytical procedures and mathematical tools used to perform quantitative analysis of PASN conductance measurements.

2.3.1 Event Identification

In the switching regime (Section 1.4.5), PASNs exhibit stepwise changes in conductance (G) in response to applied voltage. These changes in G are called *switching events* and are identified using a threshold procedure. As with all time-resolved electrical measurements, there is some level of noise from the environment in the measurement of G. The threshold procedure is used to distinguish the true changes in G caused by switching events from the apparent changes in G caused by noise.

Figure 2.7 shows a zoomed plot of a 5 kHz measurement of G (top) and the corresponding $|\Delta G|$ (bottom). $|\Delta G|$ was calculated by taking the absolute value of the point-to-point differential of G. The events denoted by the solid red markers were identified by subjecting ΔG to a threshold G_{thres} indicated by the red line. The small amplitude fluctuations in $|\Delta G|$ are due to noise and so G_{thres} is set to be slightly larger than the noise band. Any time that $|\Delta G| > G_{thres}$ an event is assigned.

[8] A third (but less relevant) advantage is the ability of the NI system to measure up to 16 channels simultaneously. This is superfluous to requirement for two-electrode PASNs, but would be advantageous when characterising multi-contact PASNs (See Section 6.3)

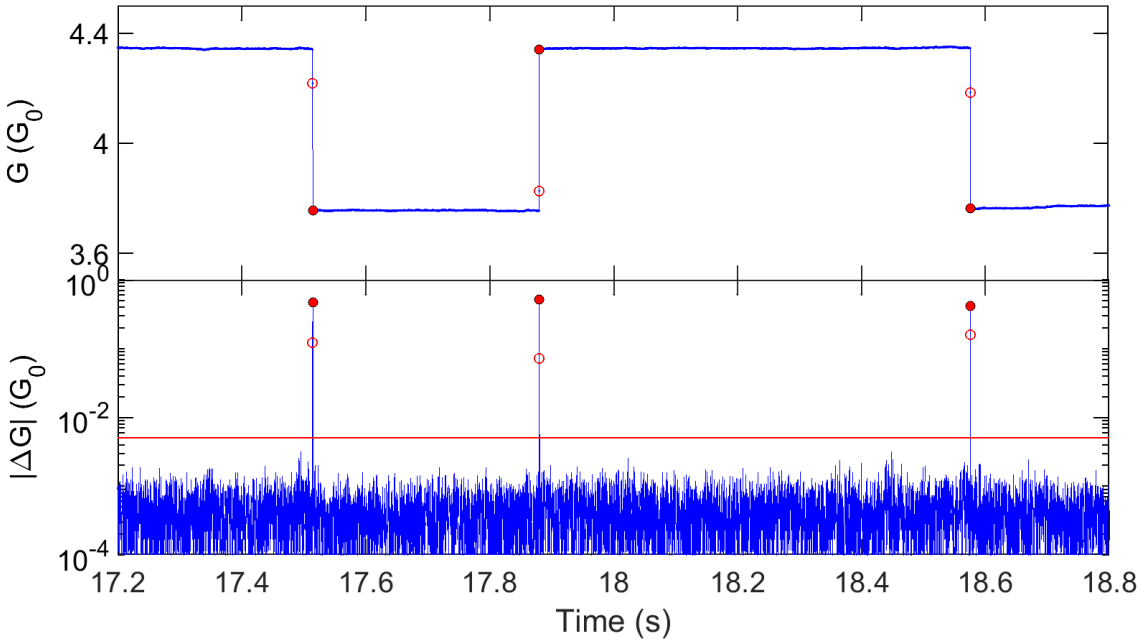


Figure 2.7: Threshold procedure. Top: A measurement of G (blue) sampled at 5 kHz using the oscilloscope measurement system. A DC bias voltage of 6 V was applied across the PASN. The solid red markers denote identified events while the open red markers indicate interstitial points. Bottom: The corresponding $|\Delta G|$(blue) calculated by taking the absolute value of the point-to-point differential of G. The red line indicates the threshold of 0.005 G_0. The same open and solid red markers are shown.

The open red symbols represent interstitial points which are a consequence of using an averaging filter during the measurement. Section 2.2.2 detailed the various measurement systems used in this thesis; each of them uses an averaging filter to minimise sampling noise. The oscilloscope and NI measurement systems both take many samples during each sampling interval and record the mean value from those samples. One consequence of this type of averaging filter is that if a stepwise change in conductance (i.e. a switching event) occurs during the sampling interval, the recorded G value will lie somewhere between the initial and final conductance values. These interstitial points create the false impression that there have been two consecutive switching events, when really there has only been one switching event. Thus, a merging procedure is used to remove the artificial events. Any two consecutive events identified using the threshold procedure (i.e. $|\Delta G| > G_{thres}$) which have the same sign in ΔG are merged together to form a single event denoted by a solid red marker at end point of the second event. The size of that event is then the sum of the two consecutive ΔG values. This procedure was used when identifying switching events from data measured using either the oscilloscope or NI systems. Data from the Keithley system, which uses a much slower

sampling rate and has a short window of averaging (20 ms) relative to the sampling interval (200 ms), is not subject to the merging procedure.

2.3.2 Probability Distributions

Probability distributions are mathematical functions which describe the probability of the different possible outcomes of an experiment. Of particular interest are power law probability distributions and exponential probability distributions which occur in many natural systems [121]. Power law distributions are often used to infer correlations, scale-invariance and criticality (Chapter 3). Exponential distributions on the other hand usually arise from uncorrelated Poisson processes [122]. Thus, it is essential in statistical analysis to be able to reliably distinguish between the two types of distribution.

2.3.2.1 Power Law Distributions

A power law distribution [121] of some variable x is given by a probability

$$P(x) = Cx^{-\alpha} \tag{5}$$

where α is some constant called the *exponent* and C is a normalization constant. C depends on whether the variable x is continuous or discrete. In the continuous case, it is required that

$$\int_{x_{min}}^{x_{max}} Cx^{-\alpha}\, dx = 1 \tag{6}$$

Which, in the limit $x_{max} \to \infty$, leads to

$$P(x) = \left(\frac{\alpha - 1}{x_{min}}\right)\left(\frac{x}{x_{min}}\right)^{-\alpha} \tag{7}$$

where $P(x)$ is the probability density function (PDF) of x, and x_{min} is the minimum value for which x follows a power law. In the discrete case it is required that

$$\sum_{x\,=\,x_{min}}^{x_{max}} Cx^{-\alpha} = 1 \tag{8}$$

and so, the PDF is given by

$$P(x) = \frac{x^{-\alpha}}{\zeta(\alpha, x_{min})} \tag{9}$$

where

$$\zeta(\alpha, x_{min}) = \sum_{n=0}^{\infty} (n + x_{min})^{-\alpha} \tag{10}$$

is the generalized zeta function. In this thesis, the power law distributed quantities of interest (e.g. inter-event interval, avalanche size etc.) are discrete and so PDFs are described using Eq.(9).

In some cases, it is also convenient to consider the complementary cumulative density function (CDF) of a power law, defined as $S(x) = \Pr(X \geq x)$[9]. For the continuous case

$$S(x) = \int_{x_{min}}^{x_{max}} P(X)\, dX = \left(\frac{x}{x_{min}}\right)^{-\alpha+1} \tag{12}$$

and for the discrete case

$$S(x) = \frac{\zeta(\alpha, x)}{\zeta(\alpha, x_{min})} \tag{13}$$

2.3.2.2 Exponential Distributions

Exponential distributions [123] describe the times between events in a Poisson point process where events occur continuously and independently at a constant average rate. An exponential distribution of some variable x is given by the PDF

$$P(x) = Ce^{-\lambda x} \tag{14}$$

where λ is the exponent and C is the normalization constant. Requiring that

$$\int_{x_{min}}^{x_{max}} Ce^{-\lambda x}\, dx = 1 \tag{15}$$

yields the normalization constant

$$C = \lambda e^{\lambda x_{min}} \tag{16}$$

The CDF of the exponential distribution is given as

$$S(x) = 1 - e^{-\lambda x} \tag{17}$$

2.3.2.3 Distinguishing Power Laws from Exponentials

Historically, a power law (exponential) distribution was identified by plotting a histogram on double (single) logarithmic axes where it appears as a straight line. The slope then represents the exponent α (λ) which could be extracted by conducting a least squares regression (Section

[9] Pr () means "probability of" and X is the observed value in x.

2.3.4.1) of the data. However, it has been shown [121] that such methods are subject to systematic errors under relatively common conditions and are thus untrustworthy. Modern techniques for identifying power law distributions in empirical data (and distinguishing them from exponential or other distributions) include using maximum likelihood estimators (MLE) (Section 2.3.4.2) and statistical tests such as the Kolmogorov-Smirnov (KS) test (Section 2.3.4.3) and Bayesian Information Criterion (BIC, Section 2.3.4.4).

2.3.3 Autocorrelation Functions

Autocorrelation is the correlation of a signal with a delayed copy of itself, as a function of the delay. Autocorrelation functions (ACFs) [124] are mathematical tools used to find repeating patterns in signals such as periodic signals obscured by noise. Formally, the ACF of a time series $y(t)$ of length N is

$$A(t_d) = \frac{\sum_{n=1}^{N-d}(y_n - \bar{y})(y_{n+d} - \bar{y})}{\sum_{n=1}^{N}(y_n - \bar{y})^2} \tag{18}$$

where $d < N - 1$. Here $\bar{y}$ denotes the mean of the time series y and t_d is the delay time with d being an integer called the *lag* which describes the number of time steps the delayed signal has been shifted relative to the original signal.

Two metrics of correlation which can be inferred from the ACF are the *correlation strength* and the *correlation range*. Correlation strength is the amplitude of the ACF taken at a particular lag value, usually $d = 1$ (i.e. $A(t_1)$ called the lag-1 autocorrelation). The correlation range is given by the maximum delay time for which $A(t_d)$ remains outside of the confidence interval Δ, where

$$\Delta \sim \pm \frac{1.96}{\sqrt{N}} \tag{19}$$

Δ represents a confidence boundary of 95%. If $A(t_d)$ lies within the confidence interval, then the signal is said to be uncorrelated. If $A(t_d)$ is initially outside the confidence interval but decays exponentially to $A(t_d) < \Delta$, then the time series is said to exhibit *short-range temporal correlation* (SRTC). If $A(t_d)$ decays slowly to $A(t_d) < \Delta$ following a power law, then the time series $y(t)$ is said to exhibit *long-range temporal correlation* (LRTC). Both

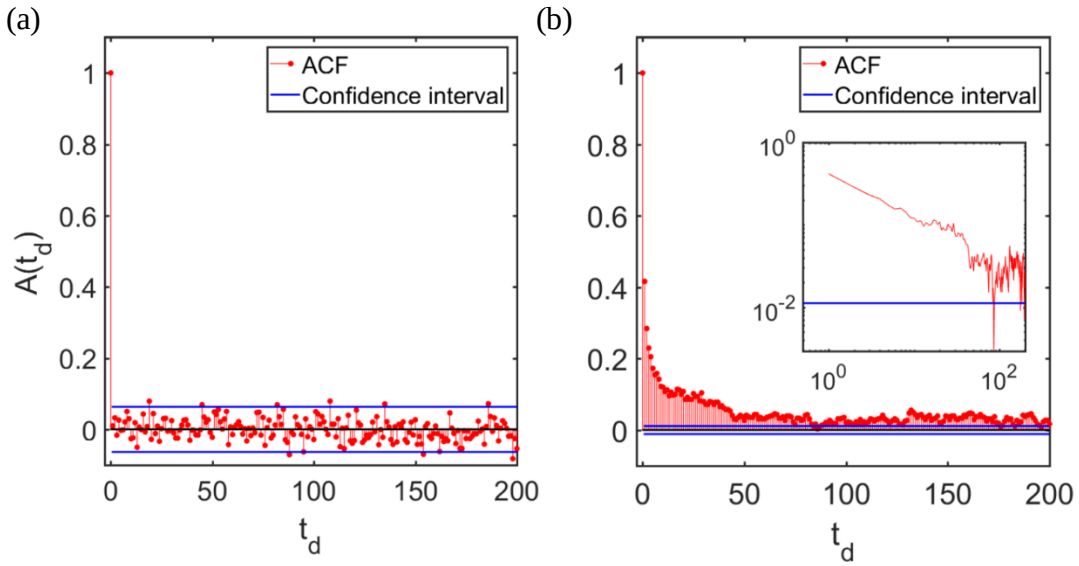


Figure 2.8: Autocorrelation function examples. (a) The ACF of an uncorrelated (random) time series. 95% of $A(t_d)$ values (red) fall within the confidence interval Δ (blue). (b) The ACF of a correlated time series shows a lag-1 correlation strength of ~0.41 and a correlation range of t_d~85. Plotting the ACF on double logarithmic axes (see inset) shows that $A(t_d)$ decays as a power law and indicates the presence of LRTC. [125]

SRTC and LRTC can be further characterised by the exponents of their respective exponential or power law ACFs.

If $y(t)$ is a random signal, then its ACF should lie within the confidence interval for all $d > 0$. The ACF for a random signal is shown in Figure 2.8 (a). $A(t_d)$ is represented by the red markers and the blue lines denoted the confidence interval. As expected, ~95% of the $A(t_d)$ values fall inside the confidence interval. If a signal is correlated, then the ACF should remain outside the confidence interval for some time. Figure 2.8 (b) shows the ACF for a correlated time series. $A(t_d) > \Delta$ until t_d~85 with a lag-1 correlation strength of ~0.41. The inset shows the ACF plotted on double logarithmic axes where it can be seen that $A(t_d)$ follows power law decay: a sign that the time series $y(t)$ features LRTC.

2.3.4 Model Fitting Procedures

This section presents several methods used to fit mathematical models to empirical data and to subsequently verify the fits of those models.

2.3.4.1 Least Squares Regression

Least squares regression [126] is a standard method for approximating the solutions of overdetermined[10] systems. It uses the sum of squared residuals[11] S as a cost function which is minimised for the optimal solution. For a model function f

$$\hat{y}_i = f(x_i, \beta) \tag{20}$$

where $\hat{y}_i$ is an estimate of the observed dependent variable y_i, x_i is an independent variable and β is a set of parameters. The least squares method minimises

$$S = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{21}$$

to determine the set of parameters β to which the observations most likely belong. The exact procedure can be found elsewhere [126,127]. In this thesis, built in MATLAB functions such as `lscov()` and `fit()` were used to perform least-squares regression. Least squares regression was used to fit the autocorrelation functions in Section 3.2.2, the $\langle S \rangle(T)$ distributions in Section 3.2.3.2 and the universal scaling functions of avalanche profiles in Section 3.2.3.3.

2.3.4.2 Maximum Likelihood Estimation

Maximum likelihood estimation (MLE) [121,128] is a method of estimating the parameters of a model probability distribution by maximising a likelihood function. Assuming a model distribution characterised by parameter(s) β, the likelihood function of a variable x of length N is defined as

$$L(\beta) = \prod_{i=1}^{N} P(x_i) \tag{22}$$

In practical applications it is often more convenient to use the log-likelihood i.e. the natural logarithm of the likelihood function:

[10] Overdetermined systems are described by more equations than there are unknown parameters.

[11] Residuals are the difference between observed values and those values predicted by a model.

$$l(\beta) = \ln\big(L(\beta)\big) = \sum_{i=1}^{N} \ln\left(P(x_i)\right) \tag{23}$$

The log-likelihood has the advantage that the product operator becomes a summation operator, allowing individual components to be maximised. Computational cost is reduced (especially when β is obtained analytically by differentiating) and the maximum of the likelihood and the log-likelihood coincide.

Maximising $l(\beta)$ yields an estimate of the parameter(s) $\hat{\beta}$ for which the observed data is most likely under the assumed model. The standard error on $\hat{\beta}$ is derived from the width of the likelihood maximum as

$$\sigma = \frac{\hat{\beta} - 1}{\sqrt{N}} + O(1/N) \tag{24}$$

Relevant to this thesis are the explicit forms of l for the power law and exponential distributions introduced in Section 2.3.2. For a discrete power law distribution with exponent α, the log-likelihood is given by

$$l(\alpha) = N \ln\big((\alpha - 1)x_{min}^{-\alpha}\big) - \alpha \sum_{i=1}^{N} \ln\left(x_i\right) \tag{25}$$

For a discrete exponential distribution with exponent λ the log-likelihood is given by,

$$l(\lambda) = N \ln\big(\lambda e^{\lambda x_{min}}\big) - \lambda \sum_{i=1}^{N} x_i \tag{26}$$

Eqs. (25) and (26) are used in Chapter 3 to fit model probability distributions to empirical data. The assumed models are verified as the best models by first using the BIC (Section 2.3.4.4) and the fitted MLE parameters are then verified by the KS test (Section 2.3.4.3).

2.3.4.3 Kolmogorov-Smirnov Test

The Kolmogorov-Smirnov goodness-of-fit test (KS test) [121,128] is method of statistically verifying a model fitted to an empirical distribution. The KS test uses an indicator called the KS distance (d_s) to compare the fitted empirical distribution with a range of synthetic distributions derived from the same model. The result is a probability (p-value) which quantifies the plausibility that the empirical distribution belongs to the fitted model.

The KS distance is defined as the maximum distance between the CDF of the model distribution $S_o(x)$ and the CDF of the empirical distribution $S_e(x)$. Formally, it is given by

$$d_s = \max|S_o(x) - S_e(x)| \tag{27}$$

in the range $x_{min} \leq x \leq x_{max}$ of the considered function. d_s is a measure of how much the empirical CDF deviates from the model CDF and is an indicator of goodness-of-fit. Smaller d_s corresponds to a better fit.

The KS distance is converted to a p-value by considering the KS distance d_s for 500 synthetic distributions $S_s(x)$ drawn randomly from the model being tested. The p-value is obtained from the fraction of the synthetic distributions that are closer to model CDF than the experimentally measured distribution.

The final result of the test is obtained by comparing the p-value with the significance value p_s. If $p > p_s$, then the hypothesis that the empirical data belongs to the model distribution cannot be rejected, and the fit passes the KS test. Conversely, if $p < p_s$, then the hypothesis is rejected, and the fit fails the KS test. In this thesis, a relatively stringent significance value of $p_s = 0.2$ was used. The KS test is used in this thesis for every fit that was performed via MLE.

2.3.4.4 Bayesian Information Criterion

To fit a model distribution to empirical data, the appropriate model must first be selected. The Bayesian information criterion (BIC) [129] is a criterion used to select a model distribution from a finite set of model distributions. The definition of BIC is

$$BIC = -2l(\hat{\beta}) + \ln(N)K_i \tag{28}$$

where $l(\hat{\beta})$ is the maximum log-likelihood, N is the number of observations in the variable of interest x, and K_i is the number of free parameters in the chosen model. The BIC is calculated for a range of candidate models and the model which produces the lowest BIC is the preferred model. The latter term appropriately adds a penalty to the BIC for models which have more parameters.

The BIC values of candidate models can be converted to a series of weights w_i which indicate the relative probabilities that each model is the most suitable. The conversion from BIC to w_i for M candidate models is given by

$$w_i = \frac{\exp\left[-(BIC_i - BIC_{min})/2\right]}{\sum_{j=1}^{M} \exp\left[-(BIC_j - BIC_{min})/2\right]} \quad (29)$$

where BIC_{min} is the smallest BIC value from the candidate models.

BIC is used in Section 3.2.3.6 to verify the selection of power law distributions against exponential, log-normal and Weibull distributions as the most suitable models for describing the distributions of the sizes and durations of avalanches in PASN switching activity. Similarly, it is used to show that the shuffled event sequences are best characterised by exponential distributions.

Chapter 3

Emergent Phenomena

Emergent phenomena [130] are properties or behaviours of a system which cannot be exhibited by the individual components of the system in isolation. They are features which emerge from the *collective interaction* of the fundamental elements of the system. An illustrative example is road traffic: a single car on a road cannot form a traffic jam, and neither can many cars on many different roads, nor can many cars using the same road at different times. Traffic jams can only occur if many cars share the same roads at the same time i.e. it is the collective interaction of many cars which cause traffic to emerge. Emergent phenomena are found in many natural systems, and life itself is considered to be an emergent property of the laws of physics and chemistry [131].

Section 1.1 introduced the brain as a biological information processing system composed of a collection of neurons which interact with one another by exchanging electro-chemical pulses via synaptic connections. Thoughts, emotions, and virtually all animal behaviour are emergent features of the brain, including the ability to perform classification/recognition/ prediction tasks which is also the goal in neuromorphic computing. Most neuromorphic approaches focus on replicating the functions of neurons and synapses on the individual level,

rather than on replicating the collective behaviour of populations of neurons and synapses [13]. While there are many novel devices which emulate neurons and synapses such as silicon-based neurons [132,133], memristor cross-bar synapses [134-136], and phase change materials [137,138], relatively little attention has been given to designing systems of functional elements which give rise to the same emergent properties which are displayed by the brain [13]. Given that the computational tasks which lie at the heart of neuromorphic computing are in fact emergent properties of the brain, it is important to emulate not only the functionality of neurons and synapses, but their collective behaviour. Furthermore, these collective behaviours can give valuable insight into the nature of the cortex and the dynamical regime which enables its impressive computational ability.

3.1 Emergent Properties of the Brain

The brain exhibits many emergent properties and functionalities. As the scope of this thesis is to elucidate whether percolating networks of nanoparticles have neuromorphic potential, focus is restricted to the emergent properties of neuronal networks which have been demonstrated via experimental investigation. The mammalian cortex is often studied both *in vitro*, where cortical slices are either grown or deposited on micro-electrode arrays [30], and *in vivo*, where living animals have electrodes surgically implanted into the brain. In both cases it has been demonstrated that neuronal populations exhibit firing patterns which have rich spatial and temporal features such as self-similar temporal dynamics and long-range temporal correlation (LRTC) [139,140], scale-free and fractal topography [29,141], hierarchical organisation [27], and critical avalanche dynamics [30,31,33]. This section introduces some of the key findings of neuroscientific studies pertaining to emergent properties of the brain.

3.1.1 Inter-event Intervals and Correlations

Inhomogeneous temporal processes often give rise to bursts of events which contain high activity periods separated by periods of quiescence [122]. There are many such processes which occur both in nature and human systems, such as earthquakes in tectonic systems [142], sunspots in solar activity [143], email sequences in digital communications [144], and

neuronal firing in the cortex [145]. These bursty sequences are often characterised by the time between consecutive events, called the *inter-event interval* (IEI). For a regular process, IEIs take on well-defined values, but the inhomogeneous processes of interest here have a wide range of IEIs, and it is the statistical distributions of IEIs which characterise the system. Systems which exhibit inhomogeneous bursts of activity often have IEI probability distributions which follow power laws of the form

$$P(t_{ie}) \sim {t_{ie}}^{-\gamma} \tag{30}$$

where P represents the probability of observing an IEI of duration t_{ie}, and γ is called the characteristic exponent. Eq. (30) says that the shorter the IEI, the more often it will be observed, but that there remains a non-zero probability of very long IEIs. Power law IEI distributions are often seen as representing "scale-free" dynamics, where there exists no characteristic time scale of the system because it is operating on all time scales [140]. The characteristic exponent γ then dictates at what rate the probability of observing a particular IEI decays with the magnitude of the IEI.

To calculate the IEI distribution of a system, its activity must be reduced to a sequence of discrete temporal events which can be interpreted as a time-dependent point process $E(t)$ [122]. In this case, $E(t) = 1$ for each time step in which an event takes place, and 0 otherwise. These binary event sequences are often referred to as *event trains*. Figure 3.1 shows temporal sequences of (a) earthquakes at a particular location, (b) neuronal firing in a rat hippocampus, and (c) outgoing mobile phone calls of an individual person [122]. The

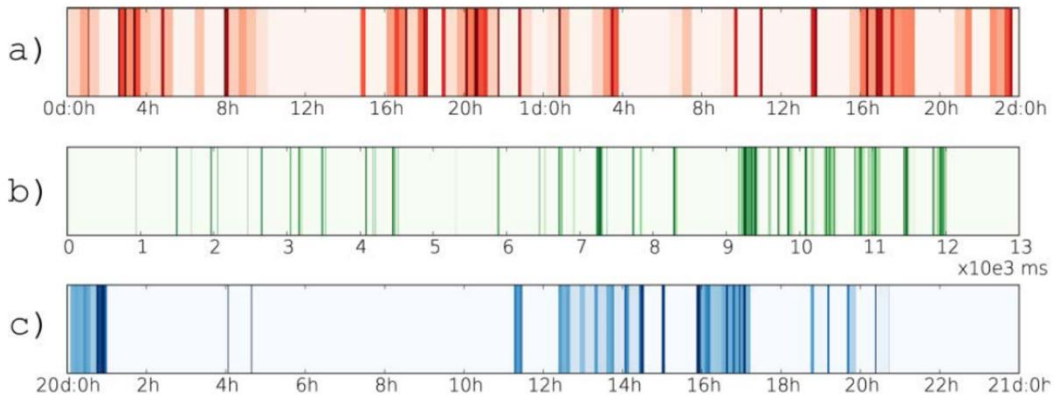


Figure 3.1: Bursty temporal sequences. Temporal sequences of (a) earthquakes with magnitude greater than 2 at a single location, (b) A single neuron firing inside a rat's hippocampus, and (c) Outgoing mobile phone calls of an individual. Shorter IEIs are denoted by darker colours. The bursty nature of the sequences is universal. Reproduced from Ref. [122] under CCL.

darker colours represent shorter IEIs and the bursty nature of the sequences are a universal emergent feature of the systems from which they originate, despite those systems being qualitatively different.

Inhomogeneous temporal processes such as those in Figure 3.1 can be characterised by the probability distributions of IEIs. Figure 3.2 shows the IEI distributions for three examples of human communications: (a) a mobile phone call sequence, (b) a sequence of short messages, and (c) an email sequence [122]. All three of the distributions feature a region (over several orders in x) which is approximately linear on logarithmic axes, indicating power law scaling behaviour. The solid lines in each plot represent regressive fits to the data from which the characteristic exponents can be estimated. For the distributions shown in Figure 3.2, it was reported that $\gamma \cong 0.7, 0.7, 1.0$ for (a), (b), and (c) respectively. Note that each of the sequences show a strong cut-off beginning at approximately $t_{ie} = 1$ day, where the probability of observing an IEI in this region decays much faster than the power law scaling observed for $t_{ie} < 1$ day. This is because human communications are dictated by daily cycles of human activity, particularly sleep patterns and typical business hours, and thus IEIs of multiple days are very unlikely.

While power law distributed IEIs are common features of correlated inhomogeneous event sequences, they alone are not sufficient to demonstrate correlation between events, as spurious power law IEI distributions may arise from uncorrelated Poisson processes e.g.

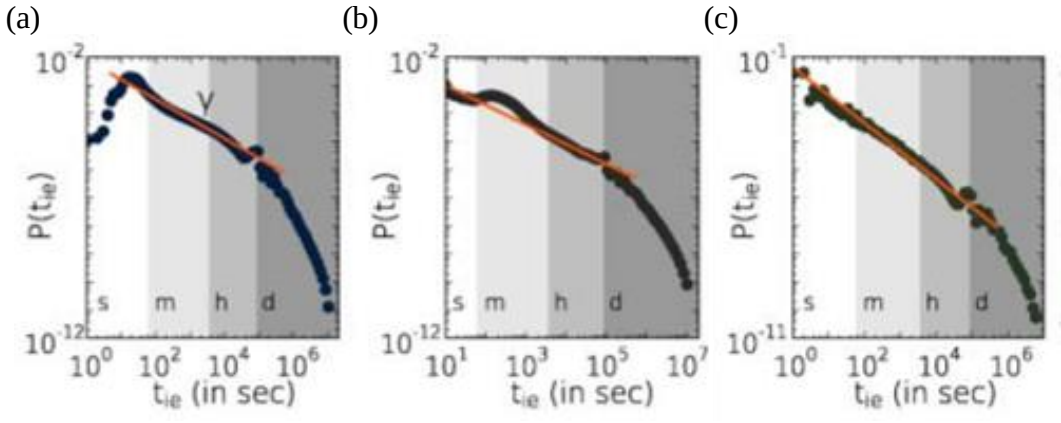


Figure 3.2: Probability distributions of IEIs in human communications. The IEI probability distributions for (a) a mobile phone call sequence, (b) a sequence of short messages, and (c) an email sequence. All three of the distributions feature a region (over several orders in x) which is approximately linear on logarithmic axes, indicating power law scaling behaviour. The solid lines in each plot represent regressive power law fits to the data yielding estimates of the characteristic exponent $\gamma \cong 0.7, 0.7, 1.0$ for (a), (b), and (c) respectively. Reproduced from Ref. [122] under CCL.

renewal processes [122,146,147]. Hence, correlation in temporal sequences of events is often characterised by the *autocorrelation function* (ACF). The ACF is the correlation of a signal with a delayed copy of itself as a function of delay. It represents the similarity between a signal at a time t and at a time $t + t_d$ where t_d is the delay time, or *lag*. A formal description of the ACF is given in Section 2.3.3.

Figure 3.3 shows the ACFs for the same temporal sequences represented by the IEI distributions in Figure 3.2 [122]. The linear regions signify power law decay (the axes are on a logarithmic scale) of the ACF, and the slopes of the fitted lines represent the characteristic exponent β, which was estimated to be 0.5, 0.6, and 0.75 for the mobile call, short message, and email sequences respectively. A smaller value of β corresponds to slower decay of the ACF amplitude and indicate that correlations persist over longer times, meaning that the occurrence of an event is correlated with events which took place further in the past. Large values of β mean that the ACF decays quickly, and that events are only correlated on short time scales [148]. The cut-offs which were observed in Figure 3.2 as a consequence of daily cycles of human activity are reflected in these ACF plots by the range over which the ACF decays as a power law. The ACF transitions from power law decay to noise at approximately the same time as the cut-offs of the IEI distributions in Figure 3.2. This is again the effect of daily cycles of human activity, where for lag values of $t_d \gtrsim 8$ hours, one would not expect significant overlap between the communication sequence and its delayed copy.

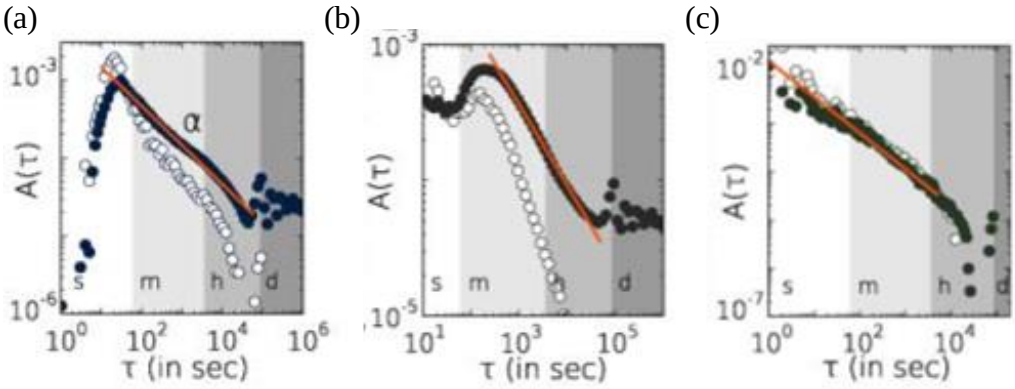


Figure 3.3: ACFs of sequences of digital human communications. ACFs of (a) a mobile phone call sequence, (b) a sequence of short messages, and (c) an email sequence. The ACF is a measure of the correlation between each event train and a delayed copy of itself, as a function of the delay τ. The linear regions represent power law decay (logarithmic axes), and the fitted curves (solid lines) can be used to estimate the ACF characteristic exponent β, which are reported as 0.5, 0.6, and 0.75 respectively. Smaller β implies slower decay in the ACF amplitude indicating a longer correlation time i.e. how long the delay must be before the ACF drops below confidence bounds (not shown here). Reproduced from Ref. [122] under CCL.

The open symbols in Figure 3.3 represent the results of the same ACF calculations on independent sequences. As the correlations in a sequence are represented by the ordering of the IEIs, shuffling the IEI sequence should destroy the correlations while preserving the IEI distribution. The shuffled distributions still produce power law decaying ACFs signifying "spurious unexpected dependencies" which the authors suggest is an illustration of the limitations of the ACF as a measure of quantifying correlations in inhomogeneous temporal processes [122]. This facilitates the requirement that the "burstiness" (i.e. avalanche dynamics) of the temporal sequences be quantified and used to unambiguously demonstrate the existence of correlations between events.

Before moving on to avalanche analysis of temporal sequences, it is worth noting that the same power law decaying IEI distributions and ACFs observed in human communications are also found in the analysis of neuronal firing patterns [122,140,149]. An example of the IEI probability distribution and ACF of the firing sequence of a single neuron is shown in Figure 3.4. The power law decaying IEI distribution (a) and ACF (b) imply scale-free dynamics and LRTC respectively. Despite the firing sequence being recorded from a single neuron, these are emergent features of the *neuronal network* in which the single neuron was embedded. The network of interconnected neurons provides the stimulus which causes the neuron being recorded to exceed its membrane potential threshold and subsequently fire an action potential. The clear power law scaling of the IEIs and ACF suggest that neuronal firing is an inhomogeneous temporal process with correlated activity occurring across multiple time scales. However, as denoted by the open symbols in Figure 3.4 (b), the independent (shuffled)

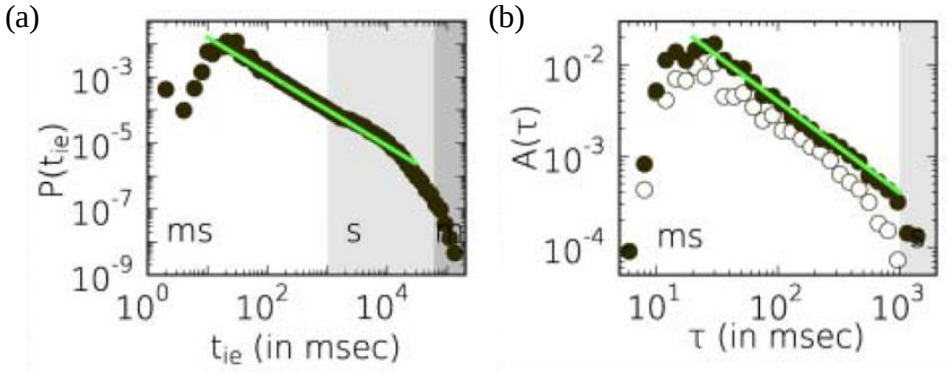


Figure 3.4: IEI distribution and ACF of a neuron firing sequence. (a) The IEI probability distribution and (b) the ACF of the firing sequence of a single neuron which exists as part of a neuronal network. The solid lines are regressive fits to the linear regions which signify power law decay over several orders in x (note the logarithmic axes). The fitted lines were used to estimate the characteristic exponents $\gamma \cong 1.1$ and $\beta \cong 2.3$. The open symbol in (b) represent the ACF of an independent sequence (see text). Reproduced from Ref. [122] under CCL.

sequences which should have no inherent correlations also appear to follow a power law decay. Hence, as in the case of human communications, characterisation of the burstiness of neuron firing sequences is required to unambiguously demonstrate LRTC.

3.1.2 Avalanche Dynamics and Criticality

Self-organised criticality (SOC) is a phenomenon where certain dissipative dynamical systems evolve to a critical point, at which the behaviour of the system becomes scale-free i.e. there exist no time or length scales which accurately characterise the system [150]. The concept is often illustrated by avalanches in a pile of sand grains [151]. As grains are dropped onto the pile one at a time, the pile grows until it reaches a critical point at which the slope of the pile fluctuates about a constant angle of repose. The addition of each new grain has the chance to trigger an avalanche on any scale relevant to the size of the pile. This model was used by Ref. [150] to demonstrate that SOC is a naturally emerging phenomena which unifies the previously unexplained but prevalent observations of "flicker" $(1/f)$ noise [152] and spatial self-similarity i.e. fractal structures [153] in a wide range of different dynamical systems.

The term "avalanche" was originally used to describe the sudden cascade of sand grains in a pile, but its meaning has since been extended to refer to any burst of activity in any self-organised critical system. For example, an earthquake may trigger an avalanche of aftershocks in a tectonic system [154], or an action potential may trigger an avalanche of neuronal firings in a cortical network [30]. This section describes some key results from the literature on avalanches and criticality in human communications, and in the brain.

As noted in Section 3.1.1, Ref. [122] posits that "heavy-tailed" IEI distributions and power law decaying ACFs are not sufficient methods of fully quantifying temporally correlated heterogeneous behaviour due to spurious dependencies observed in uncorrelated sequences. Ref. [122] instead suggests that it is rather the distribution of the number of events in a bursty period which serves as the best indicator of dependencies. The same binarized sequences (event trains) of mobile phone calls, short messages and email sequences from Section 3.1.1 were analysed for avalanches, using the string method for burst identification [155]. This involves choosing a maximum IEI (t_{max}) which may occur within a burst and arguing that any two events which occur within that interval are correlated due to their close temporal proximity. It then follows that a burst is any sequence of successive events which

are separated by IEIs $< t_{max}$, and that any event which occurs within a burst is dependent on the occurrence of the preceding events within that burst. The burst size probability distribution $P(E)$ is generated by counting the number of events E in each burst. If the sequence is uncorrelated i.e. the events occur independently of one another, then $P(E)$ is uniquely determined by the IEI distribution $P(t_{ie})$ such that

$$P(E = n) = \left(\int_0^{t_{max}} P(t_{ie})dt_{ie}\right)^{n-1}\left(1 - \int_0^{t_{max}} P(t_{ie})dt_{ie}\right) \qquad (31)$$

The integral inside the first parentheses denotes the probability of randomly drawing an IEI $\leq t_{max}$, therefore the first term is the probability that this is done independently $n-1$ consecutive times. The second term gives the probability that the n^{th} draw is an IEI $> t_{max}$, and so the number of events in the burst $E = n$. Because t_{max} is finite, the integral $\int_0^{t_{max}} P(t_{ie})dt_{ie} = a$ where the constant $a < 1$, resulting in asymptotic behaviour of the general exponential form $P(E = n) \sim a^{n-1}$. Consequently, any finite sequence of uncorrelated events will have a $P(E)$ distribution which decays *exponentially*, even if the $P(t_{ie})$ distribution follows a power law. Any deviation from the exponential decay is indicative of correlations in the timing of successive events [122].

Figure 3.5 shows the resulting $P(E)$ distributions for (a) a mobile phone call sequence, (b) a sequence of short messages, and (c) an email sequence. These are the exact same sequences used to produce the $P(t_{ie})$ distributions and ACFs in Figure 3.2 and Figure 3.3 respectively. The $P(E)$ distributions are all heavy-tailed i.e. they are approximately linear on

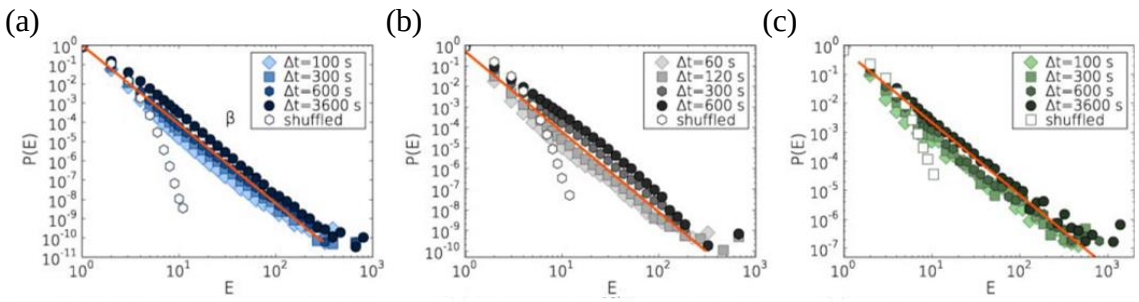


Figure 3.5: Burst size distributions for digital communications sequences. Probability distributions P of the number of events in a burst E for (a) a mobile phone call sequence, (b) a sequence of short messages, and (c) an email sequence. The $P(E)$ distributions are heavy-tailed, regardless of the bin size t_{max} used in the calculation (see text). Note that Ref. [122] uses the symbol Δt instead of t_{max} in the legends. Shuffling the sequences destroys the correlations between event times and their $P(E)$ distributions (open symbols) are consequently exponential. Reproduced from Ref. [122] under CCL.

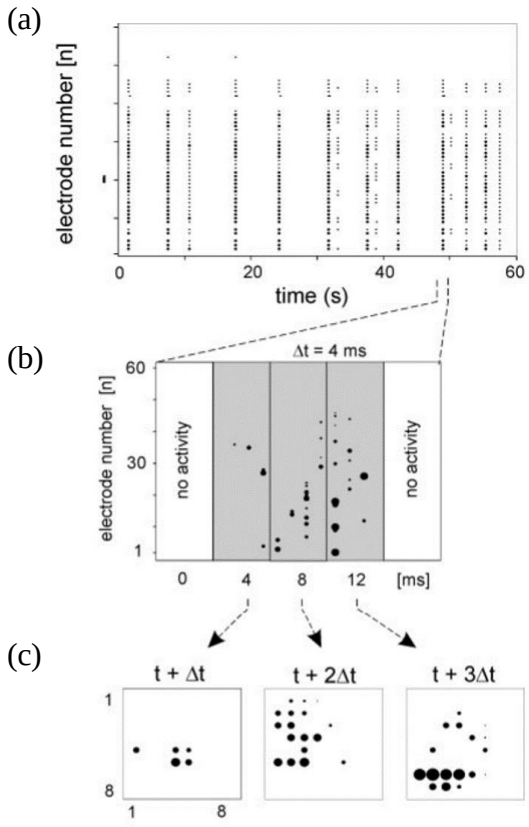


Figure 3.6: Definition of a neocortical avalanche. (a) Subsection of a 60-channel MEA recording of spontaneous neuronal activity showing (b) correlated periods containing spatiotemporal patterns. Avalanches were defined as sequences of continuous activity which were preceded and terminated by empty frames. (c) the spatial distribution of neuronal firings during a 3-frame avalanche (dot sizes represent magnitude of recorded LFPs. Reproduced from Ref. [30] under CCL.

logarithmically scaled axes implying power law scaling (but a more rigorous fitting procedure would be required to confirm true power law dependency). In any case, the $P(E)$ distributions certainly do not decay exponentially, as would be the case for an uncorrelated sequence of events. Ref. [122] correctly identified the ambiguity in the choice of t_{max} which determines how an avalanche is defined. Hence, they show the $P(E)$ distributions for a wide range of t_{max} values[12], thus demonstrating the invariance of the heavy-tailed distributions to the choice of t_{max}: although the distributions do change slightly, the distributions remain heavy-tailed and their slopes remain similar. In stark contrast are the $P(E)$ distributions of the shuffled sequences, denoted by the open symbols in Figure 3.5. The shuffled $P(E)$ distributions drop away much faster than any of the unshuffled distributions and are indeed representative of exponential decay, in accordance with Eq. (31). The merits of

[12] Note that Ref. [122] uses the symbol Δt instead of t_{max} in the legends of Figure 3.5.

preferentially using the burst size distribution over the ACF as a method of identifying correlations is readily obvious upon comparison of Figure 3.3 and Figure 3.5. While the ACFs in Figure 3.3 do show LRTC represented by the slow power law decay, the shuffled sequences also exhibit this behaviour. On the other hand, the exponential $P(E)$ distributions of the shuffled sequences are distinctly different than the heavy-tailed $P(E)$ distributions of the unshuffled sequences signalling that the original event sequences were in fact correlated.

3.1.2.1 Avalanche Size and Duration

A procedure similar to the analysis of bursty time series outlined above was used in Ref. [30] to demonstrate the existence of avalanches in cortical networks. They investigated both mature organotypic cultures and acute slices of rat cortex by recording the local field potentials (LFPs) produced by neurons firing, using 60 channel multielectrode arrays (MEAs). They define a *frame* as the spatial pattern of electrodes which recorded neurons firing during a time bin of width Δt. An avalanche (equivalent to a burst) was then defined as any number of consecutive frames featuring neuronal activity, which were both preceded and terminated by a blank frame (no activity). This definition is illustrated pictorially in Figure 3.6 [30], where the bin size $\Delta t = 4$ms is equivalent to the mean value of the IEI distribution, ⟨IEI⟩. Figure 3.6 (a) shows a recording of spontaneous neuronal activity that features avalanches of neuronal firing. An example of an avalanche is shown in Figure 3.6 (b), while Figure 3.6 (c) demonstrates that the spiking events are distributed across many of the MEA electrodes indicating spatiotemporal correlation within the cortical network. Ref. [30] further defines the *avalanche size* (S) as the total number of recorded LFPs which occur in a single avalanche, and the *avalanche duration* (T) as the number of frames over which the avalanche occurs. This definition of avalanche size is consistent with the definition of E used in Ref. [122] in the analysis of bursty communications sequences. Similarly, it was found that the probability distributions of avalanche size $P(S)$ are heavy-tailed and follow power law decay over several orders of magnitude, as shown in Figure 3.7 (a). The power law decay is expressed as

$$P(S) \sim S^{-\tau} \tag{32}$$

where $\tau \approx 1.5$ for a bin size of $\Delta t = \langle IEI \rangle$.

As the bin size Δt is a free parameter, the analysis was repeated for different bin sizes ranging from 0.25 - 4 times $\langle IEI \rangle$ and it was found that the $P(S)$ distibutions follow power

law decay independently of the bin size used, though the power law exponent τ is dependent on the bin size, as depicted by the inset in Figure 3.7 (a). The exponent for avalanche size τ was estimated to be approximately 3/2, which is in agreement with the predictions of a critical branching process [156] (discussed further in Section 3.1.2.2). The dependence of τ on Δt comes from the fact that longer bin sizes can result in several shorter avalanches being concatenated into one single avalanche. This leads to many fewer short avalanches and

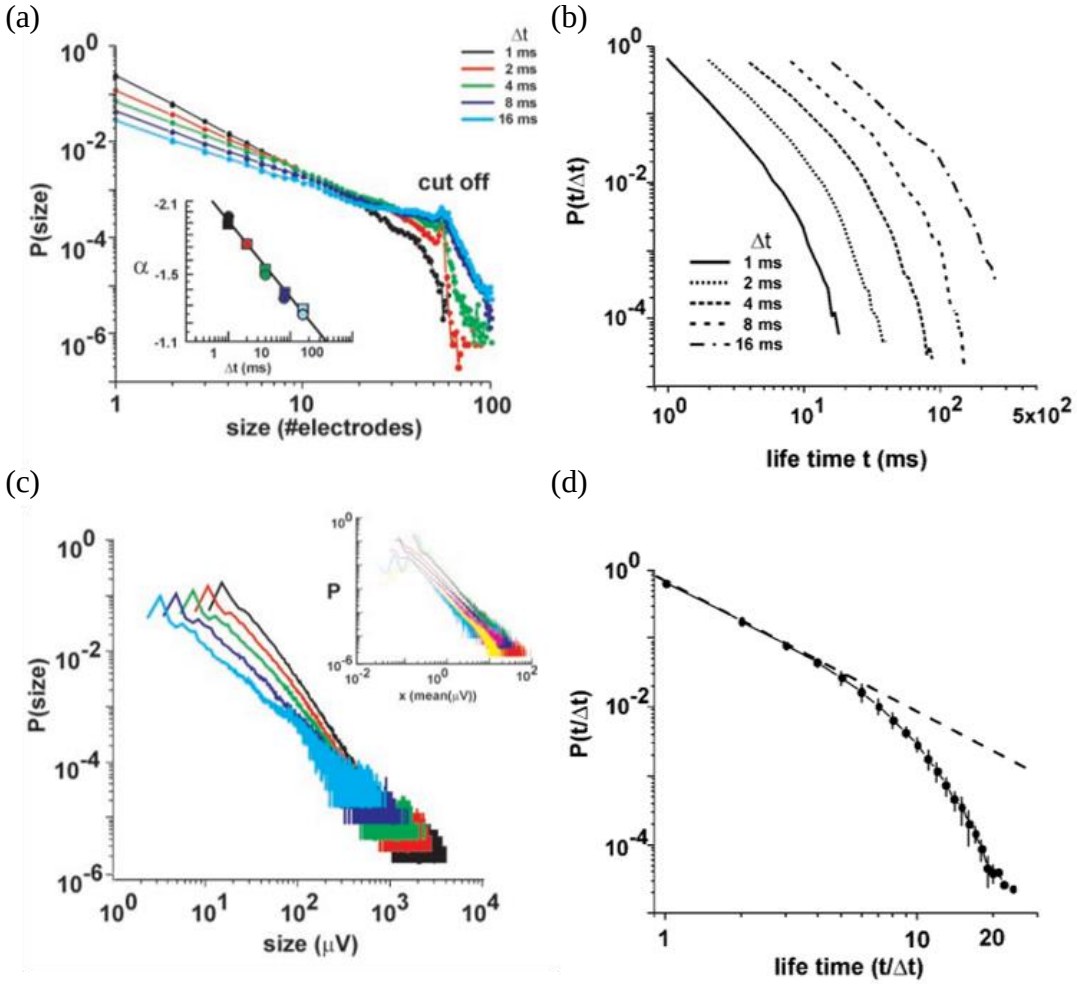


Figure 3.7: Avalanche size and duration distributions from biological neuronal networks. (a) Probability distribution of avalanche sizes (number of LFPs recorded) in log-log coordinates for different Δt (average for seven cultures). The linear regions indicate power law dependency $P(S) \sim S^{\tau}$. Cut-off induced by maximum number of electrodes ($n \approx 60$). Inset shows dependence of size exponent τ on Δt also follows a power law with exponent -0.16 ± 0.01. (b) Duration distributions of cortical avalanches follow a power law $P(T) \sim T^{\alpha}$ ($\alpha \approx -2$) with an exponential cut-off consistent with finite size effects. (c) Probability distribution of avalanche size based on summed LFPs as a function of bin width Δt. Inset is an overplot of power laws for all seven cultures at $\Delta t = 1$ms expressed in multiples of average LFP magnitude. (d) Normalized time $t/\Delta t$ gives scale-free duration distribution (average over $\Delta t = 1$, 2, 4, 8, 16 ms for seven cultures). Dashed line represents a slope of -2. Reproduced from Ref. [30] under CCL.

several more large avalanches, corresponding to a decrease in the slope of $P(S)$. The opposite is true for shorter bin sizes, where large avalanches may be broken into several smaller avalanches. Interestingly, it was found that the variation of τ with Δt also follows a power law $\tau(\Delta t) \sim \Delta t^{-0.16 \pm 0.01}$. The strong cut-off in Figure 3.7 (a) observed at $S \approx 60$ is due to the number of electrodes on the MEA: because neurons have a refractory period following the release of an action potential [157], it is unlikely that the same neurons fire more than once during an avalanche, essentially setting an upper limit on the number of LFPs which can be observed.

Similar results to those displayed in Figure 3.7 (a) were found when the avalanche size definition was modified to include the amplitude of the recorded local field potentials, as shown in Figure 3.7 (b). In this case, the avalanche size is expressed as the absolute sum of LFP peak amplitudes and continues to display power law decay regardless of the chosen Δt, and again a slope of $\tau \approx 1.5$ was observed. The inset of Figure 3.7 (b) shows the same result for seven different cortical networks binned at $\Delta t = 1$ms and expressed in terms of the mean LFP magnitude. This shows that at any given scale, the ratio of patterns with size above or below that scale is constant, leading to the conclusion that the dynamics do not have a critical size threshold i.e. cortical networks exhibit scale free dynamics.

The distribution of avalanche durations is another important parameter in characterising system dynamics, describing the temporal dimension of propagation. Ref. [30] shows that, in accordance with theoretical considerations [156], and neuronal network simulations [158], the probability distribution of the durations of cortical avalanches $P(T)$ also follows a power law, but with an exponential cut-off. The power law governing the distribution of avalanche durations is

$$P(T) \sim T^{-\alpha} \tag{33}$$

These results are shown in Figure 3.7 (c), where $P(T)$ is shown as a function of Δt, and in Figure 3.7 (d) where the transformation $t' = t/\Delta t$ was used to demonstrate that the distributions in (c) collapse onto a single scale-invariant duration distribution which has an initial slope of $\alpha \approx 2$.

3.1.2.2 The Mechanism behind Cortical Avalanches

Section 3.1.2.1 described the results of Ref. [30] which presented the first conclusive demonstration of power law distributed avalanches in cortical networks. The inquisitive reader will no doubt be wondering what the generative mechanism behind these avalanches is, and whether there is any physical meaning to the determined power law exponents. Ref. [30] theorises that these observations are the result of a self-organised critical branching (SOCB) process [156,159].

A branching process can be described in the context of mean-field theory [150] where the sites $\boldsymbol{r}$ of a d-dimensional lattice are assigned a binary value $z(\boldsymbol{r})$ (i.e. a cellular automaton) which are allowed to change according to a fixed rule. A site $\boldsymbol{r}_i$ can be described as 'active' if $z(\boldsymbol{r}_i) = 1$, and 'inactive' if $z(\boldsymbol{r}_i) = 0$. An inactive site may then become active in the next time step with probability $1 - h$, if at least one of its neighbours (an adjacent site) is active, and with probability h if no neighbours are active. This is called a branching process because each newly activated site has the chance to trigger further activation, or die out [160], and can result in the propagation of avalanches (fronts of site activation). Mean field theory has been successfully used to describe a number of physical phenomena, including the Ising model for ferromagnetism [161], and the widely studied sand pile models of SOC [150], which are in fact characterised by a *critical* branching process [156].

For a branching process to be critical, some control parameter must be fine-tuned to a critical value; in the mean field example, that parameter is h, the probability that the neighbour of an active site becomes active in the next time step. But the nature of self-organised systems such as sand piles prohibits this fine tuning because the system evolves to a stationary state naturally. The SOCB process reconciles this paradox by coupling local dynamical rules with global conditions which regulate the total energy of the system and produce a critical branching process without the need to fine tune the control parameter. The SOCB process has been shown to produce scale-free avalanches which have power law probability distributions [156] consistent with the values of $\tau = -3/2$ and $\alpha = -2$ found in mean field theory [150] and in the analysis of avalanches in neuronal networks [30].

Ref. [30] investigated the presence of a branching process in cortical networks by directly calculating the branching parameter σ [162]. σ gives the expected number of MEA electrodes that will be active in the next time step following the activation of a single electrode. The average number of descendant electrodes can be approximated by the ratio of descendent

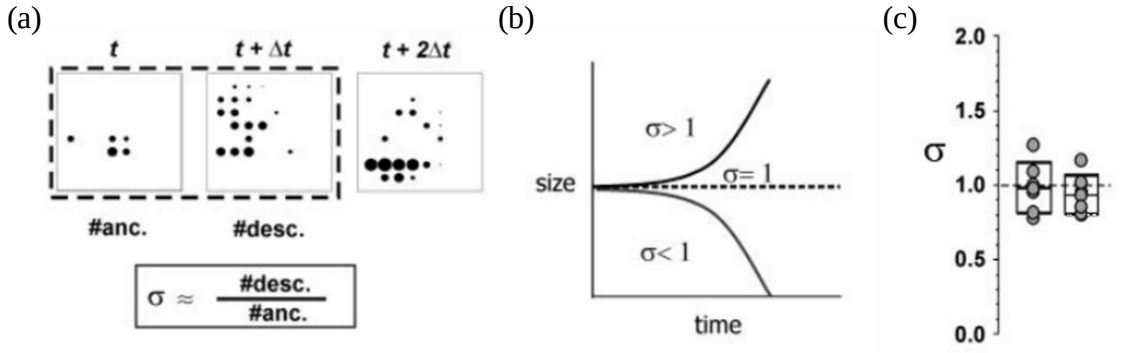


Figure 3.8: Critical branching parameter in neuronal networks. (a) Illustration of the branching parameter σ as the ratio of the number of activated MEA electrodes in the first and second frames of an avalanche. (b) For $\sigma > 1$, avalanche size diverges taking over the whole network, while for $\sigma < 1$ the avalanche dies away quickly. At the critical value $\sigma = 1$, avalanches persist at all scales. (c) Values of σ calculated from cortical avalanches with (left) one active electrode and (right) multiple active electrodes in the first frame. The values are in agreement with a critical branching process as the mechanism behind cortical avalanches. Reproduced from Ref. [30] under CCL.

electrodes to ancestor electrodes for two consecutive time bins at the beginning of an avalanche, as depicted in Figure 3.8 (a) [30]. If $\sigma > 1$, then the system is increasingly activated at each time step, leading to a runaway (supercritical) state, and if $\sigma < 1$ then activity decreases at subsequent time steps and the system quickly returns to a (subcritical) state of dormancy. If $\sigma = 1$, then the branching process is critical, and on average, the activation of n electrodes at one time step leads to the activation of n electrodes in the next time step, keeping the network at the edge of stability (depicted in Figure 3.8 (b)). Ref. [30] reports values of σ which are remarkably close to the critical value of 1: they find $\sigma = 1.04 \pm 0.19$ for avalanches starting with one active electrode, and $\sigma = 0.90 \pm 0.19$ for those avalanches beginning with more than one activated electrode, as shown in Figure 3.8 (c). To rule out chance occurrence, they repeated the analysis for 50 control sets which had the event times jittered by 4-80 ms. For jitter of $\pm$ 4ms, σ had decreased significantly to 0.7 and decreased further with increased jitter. This demonstrates that the original branching parameter was significantly different than that expected by chance. The realization of a critical branching parameter, as well as the power law distributed avalanche properties (size, duration) with exponents $\tau = -3/2$ and $\alpha = -2$, are together strong evidence for a critical branching process as the mechanism responsible for avalanches in cortical networks.

3.1.2.3 Criticality Beyond Power Law Scaling

The evidence for criticality in the brain outlined in Sections 3.1.2.1 and 3.1.2.2 relies heavily on the presence of power law scaling in the statistics of avalanches in the measured LFPs of cortical cultures [30]. The inference of such strong conclusions from the presence of apparent power law distributions has been appropriately cautioned for several reasons [163,164]. One reason is because power law distributed avalanches predicted by theory [150,156] arise from the consideration of infinite systems, while empirical studies examine finite systems and datasets [164]. It has also been noted that power law statistics may result from uncorrelated processes with origins unrelated to criticality or a second order phase transition [165]. For example, it has been shown that a random typewriter would generate texts that have word frequencies distributed as a power law [166] which demonstrates that such statistics may arise from purely stochastic mechanisms. Furthermore, it has been shown that apparent power law distributions may arise from combinations of exponential distributions [167], and that the inverse of regularly distributed quantities may be power law distributed [168]. Other examples of power law scaling away from criticality include random walks [165], and systems with aggregation and injection [169], as well as models of neuronal networks which lie in the regime of balanced inhibition and excitation [170].

The presence of power law scaling in avalanche statistics alone is clearly an insufficient measure of demonstrating criticality, and more thorough analyses must be performed. This motivated the work of Ref. [31] where high resolution measurements of the spiking of *in vitro* neuronal cultures were shown to conform to several aspects of the unified theory of critical systems [171].

Ref. [171] uses renormalization group methods to demonstrate that the underlying cause of universal scale-free behaviour in critical systems is a fixed point in system space at which models map to the same point under coarse-graining, resulting in similar behaviour at all scales. This theory predicts that, in accompaniment to the previously demonstrated power law distributed avalanche sizes and durations, the average size of avalanches of a particular duration should also scale as a power law as a function of duration. This notion is based on the observation that in magnetic systems, larger avalanches (with more domain flips) take longer to finish and have internal self-similar statistics (e.g. Figure 3.10), and provides a relationship between the spatial and temporal domains of avalanche dynamics. The derivation is as follows [171]. All avalanches of a duration T will have a distribution of sizes

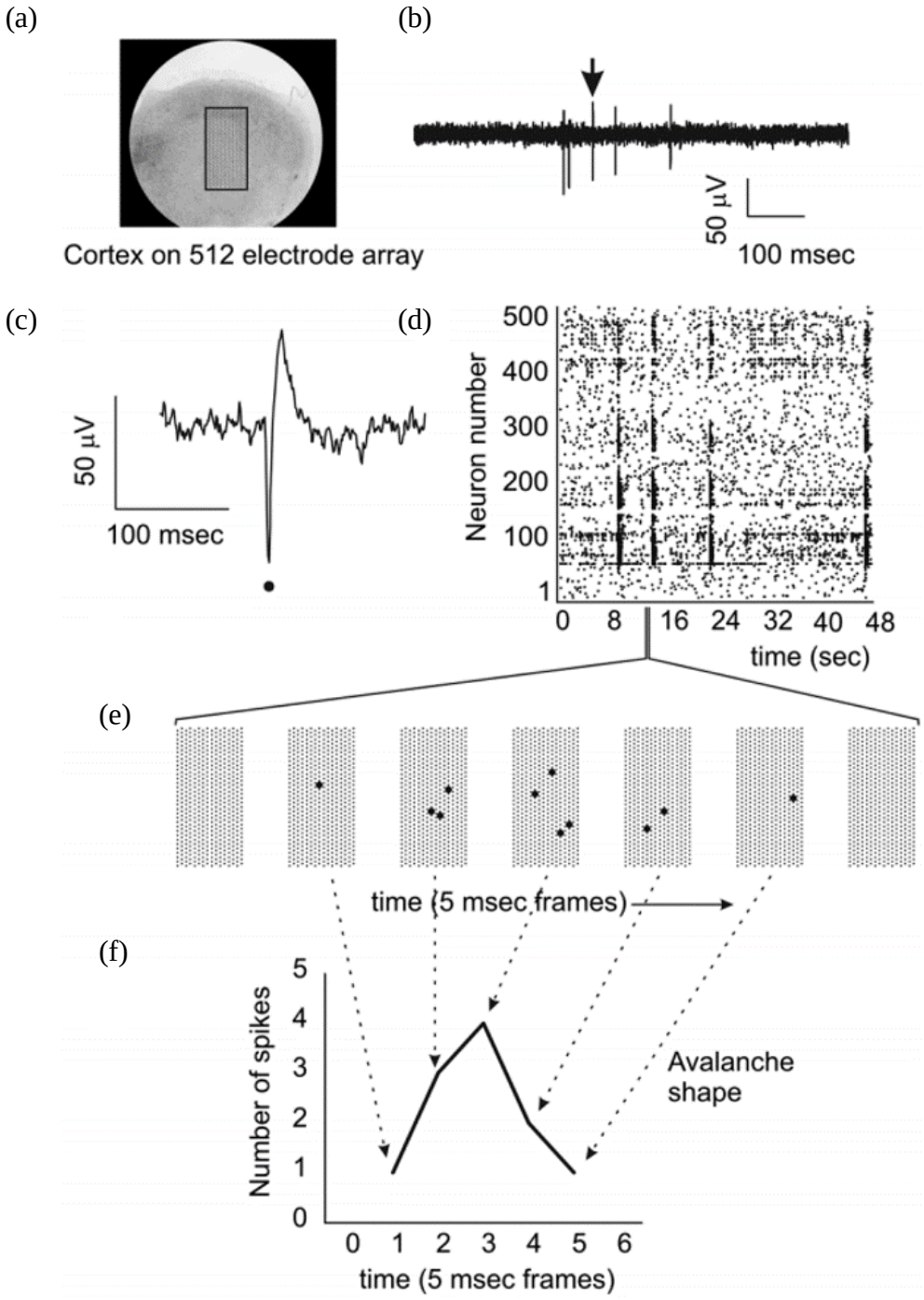


Figure 3.9: Avalanche profile extracted from MEA recordings of cortical tissue. (a) A micrograph of a cortical slice on a 512 electrode MEA (black rectangle denotes array location). (b) Voltage trace from one electrode showing spikes from neurons firing. (c) Time of spike marked as black dot. (d) Raster plot of spike times from many neurons over a 48 s interval. (e) An avalanche represented as a series of frames: each frame is the entire array (electrodes are small dots) with the large dots denoting spikes which occurred during a 5 ms interval (Δt). An avalanche is a series of consecutively active frames with inactive frames either side. (f) The avalanche shape (temporal profile) is obtained by plotting the number of spikes in each frame. Reproduced with permission from Ref. [31].

with mean $\langle S \rangle$. $\langle S \rangle(T)$ can be compared to itself on a slightly longer time scale (as in Ref. [172]) by scaling time by a small factor of $1 - \delta$. This should lead to a correspondingly small rescaling of the average size by a factor $1 + a\delta$, such that

$$\langle S \rangle(T) = (1 + a\delta)\langle S \rangle\big((1 - \delta)T\big). \tag{34}$$

In the limit $\delta \to 0$, this becomes

$$a\langle S \rangle = T\frac{d\langle S \rangle}{dT} \tag{35}$$

the solution of which is the power law relation

$$\langle S \rangle(T) = S_0 T^a \tag{36}$$

where the exponent a is called the critical exponent. Ref. [171] renames a in terms of three other critical exponents, σ, ν, and z, such that $a = 1/\sigma\nu z$. This convention is adopted in this thesis for consistency with the literature, but the descriptions of the parameters σ, ν, and z are not pertinent here and can instead be found in Ref. [171] and citations therein.

Another prediction of the unified theory of critical systems [171] is the collapse of avalanche shapes onto a universal scaling function. Consider the event train $E(t)$ of a critical system undergoing avalanches. The *event rate* $s(t)$ can be obtained by binning $E(t)$ in the time domain giving the number of events per unit time. The nature of avalanche dynamics

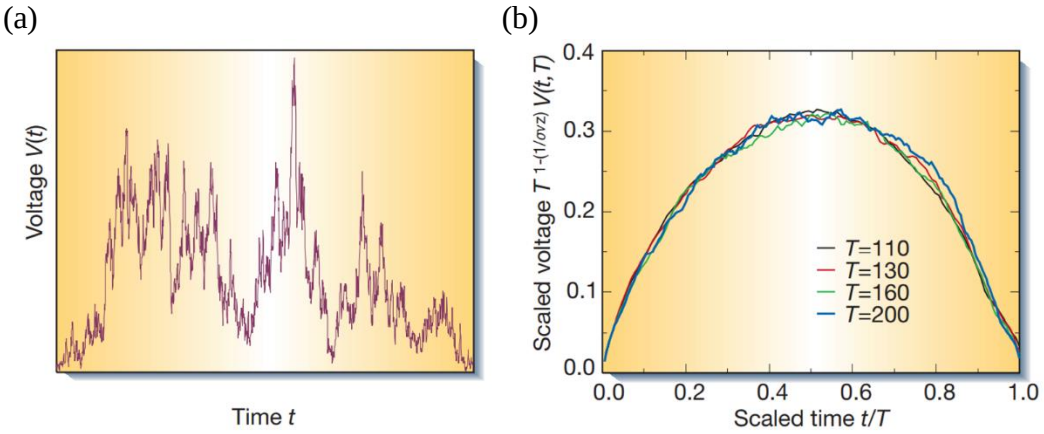


Figure 3.10: Avalanche shape collapse from Barkhausen noise. (a) Voltage pulse representing the number of magnetic domains flipped per unit time during a single large avalanche. The avalanche is noisy, almost dying several times necessitating that shapes be averaged over all occurrences of the same duration. (b) Collapse of the average avalanche shapes of 4 different large durations onto a single scaling function which is approximately parabolic. Shape collapse provides a more robust assessment of criticality than independent size and duration scaling. Reproduced with permission from Ref. [171].

dictates that there are large fluctuations in $s(t)$ which are separated by periods where $s(t) = 0$ (quiescence). The shape of an avalanche (also called the temporal profile) is then defined as the set of non-zero $s(t)$ values which lie between the preceding and trailing empty bins, as per the definition of an avalanche given in Section 3.1.2.1. This is illustrated in Figure 3.9 [31] where the MEA (a) recordings of neuronal spiking (b, c) are resolved into 'frames' (equivalent to time bins with spatial information) (d, e) which are used to plot the temporal profile of the avalanche (f). As avalanches typically contain large fluctuations, such as in the example of Barkhausen noise shown in Figure 3.10 (a) [171], the avalanche shapes are averaged over all avalanches of equal duration to produce the *mean temporal profile* for each unique duration. Let $\langle s \rangle(t, T)$ denote the mean temporal profile for avalanches of duration T. A change of variables $t \to t/T$ allows $\langle s \rangle(t, T)$ to be compared to itself on a time scale shifted by a small factor $1 - \delta$ such that

$$\langle s \rangle(t/T, T) = (1 + b\delta)\langle s \rangle(t/T, (1 - \delta)T) \tag{37}$$

As $\delta \to 0$, Eq. (37) becomes

$$b\langle s \rangle = T\frac{\partial \langle s \rangle}{\partial T} \tag{38}$$

Which is solved by

$$\langle s \rangle = s_0 T^b \tag{39}$$

Because the integration constant depends on t/T, $s_0 = \boldsymbol{s}(t/T)$, the final scaling form is given by

$$\langle s \rangle(t, T) = T^b \, \boldsymbol{s}(t/T) \tag{40}$$

where the scaling function $\boldsymbol{s}(t/T)$ is a universal prediction of the theory [171].

The mean temporal profiles $\langle s \rangle(t, T)$ of each unique duration T should collapse onto the scaling function $\boldsymbol{s}(t/T)$. This is illustrated in Figure 3.10 (b) by plotting $T^{-b}\langle s \rangle(t, T)$ against t/T for 4 long avalanche shapes from experimentally measured Barkhausen noise [173]. All of the plots fall onto the same curve (which is approximately parabolic) clearly indicating self-similarity. Ref. [171] cautions that if the shape collapse yields plots which do not all look alike, then any power laws measured are likely accidental. The exponent b provides an additional method of estimating the critical exponent $1/\sigma\nu z$: because $\langle S \rangle(T) = \int \langle s \rangle(t, T)dt = \int T^b \, \boldsymbol{s}(t/T) \sim T^{b+1}$ must equate with the scaling relation given by Eq. (36), it is clear that $a = b + 1 \to b = 1/\sigma\nu z - 1$.

The scaling of the average size of avalanches with their durations, as well as the collapse of avalanche profiles onto a single function are considered to be much more robust measures of demonstrating criticality than the mere presence of power law distributed avalanche sizes and durations [174,175]. However, in a model of neuronal networks based on Boltzmann's theory of molecular chaos, it has been demonstrated that both size-duration scaling and avalanche shape invariance can arise by purely stochastic mechanisms [175]. This necessitates an additional criterion when demonstrating criticality in dynamical systems: satisfaction of the relationship between exponents known as the "crackling relationship" [171,176] which unifies the exponents of avalanche size (τ) and avalanche duration (α) such that

$$\frac{1}{\sigma\nu z} = \frac{\alpha - 1}{\tau - 1} \tag{41}$$

There are thus three independent measures of obtaining an estimate of the critical exponent $1/\sigma\nu z$:

1. By measuring the power law exponents of avalanche size and duration and using Eq. (41).
2. By measuring the power law exponent of the average size given duration $\langle S \rangle(T)$.
3. By performing a shape collapse onto a known scaling function.

Although the models of Ref. [175] could reproduce the power law distributed avalanche statistics and shape invariance, they could not satisfy the crackling relation, which is therefore recommended by the authors as a method of distinguishing systems which are truly critical from those which may give spurious signs of criticality. Hence, the agreement of these three independent estimates of the critical exponent $1/\sigma\nu z$ is crucial for a robust demonstration of criticality consistent with the theory of [171].

The results from Ref. [31] indeed satisfy the criteria outlined in Ref. [171]. Figure 3.11 shows histograms of avalanche size and duration in log-log coordinates for (left column) a critical cortical network, and (right column) a subcritical cortical network [31]. The size and duration distributions for the critical case have linear regions over several orders of magnitude, indicating power law scaling consistent with Eqs. (32) and (33) respectively. The corresponding power law exponents were reported to be $\tau = 1.7 \pm 0.2$ and $\alpha = 1.9 \pm 0.2$. The bottom row of Figure 3.11 contains the average size given duration which scales in accordance with Eq. (36) in both the critical and subcritical cases; though the range of data from the subcritical network is limited to < 2 orders reflecting the early cut-off in the duration

distribution as compared with the critical case. The critical exponent is reported as $1/\sigma\nu z = 1.3 \pm 0.05$. These three exponent values do satisfy the crackling relationship (Eq. (41)) within their uncertainties.

Finally, Figure 3.12 shows the avalanche shape collapse (bottom row) of three mean temporal profiles (top row) of long duration from the same two samples as in Figure 3.11.

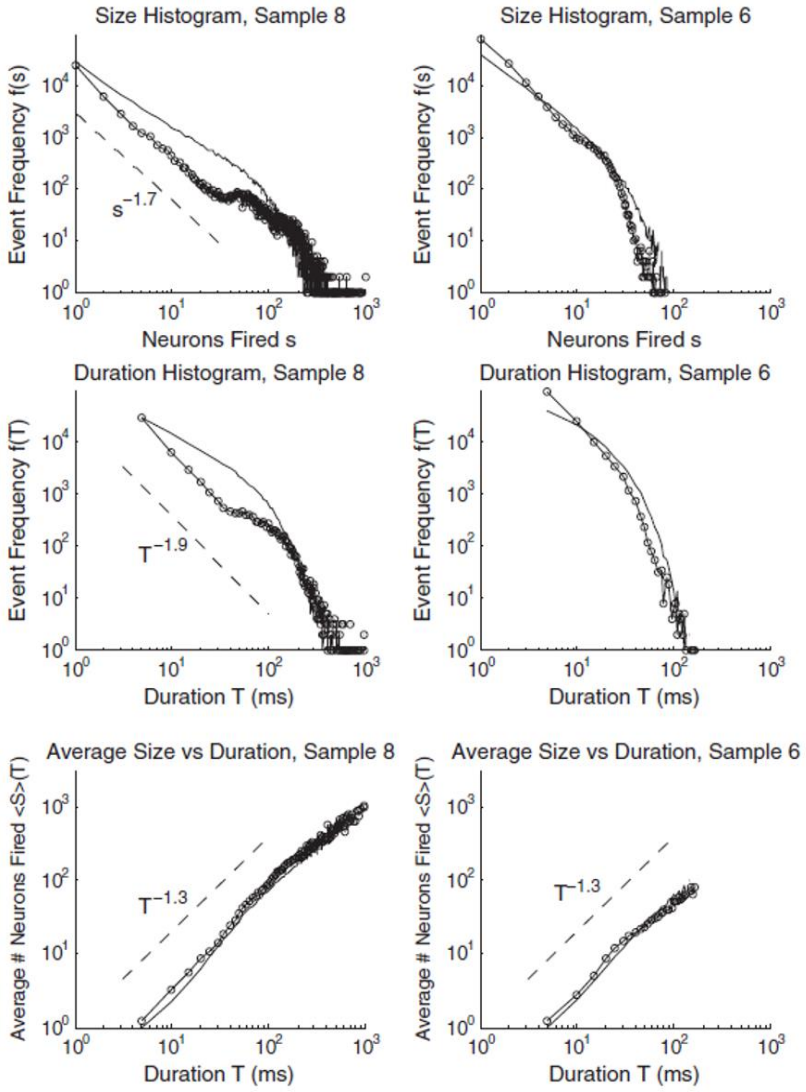


Figure 3.11: Avalanches in cortical networks follow power law scaling. Contrasting size and duration distributions and average size for fixed duration from a critical data set (left) and a subcritical data set (right). Experimental data are shown by lines with markers and data from the corresponding models are shown by smooth lines. The dashed lines on the left (near critical) column correspond to power laws with exponent 1.7, 1.9, and 1.3, corresponding to the critical exponents τ, α and $1/\sigma\nu z$ respectively. These values satisfy the exponent relation (Eq. (41)) as is expected for a system near criticality. Statistical error bars are only significant for the largest and longest events for the experimental data and are too small to see in the figure for the simulations. Reproduced with permission from Ref. [31].

The left column is from the critical network and shows a clean collapse of the avalanche shapes onto a scaling function which is approximately parabolic. This behaviour is reproduced by a simulation of the critical cortical network (middle column, see [31] for details). The subcritical network (right column) does not exhibit shape collapse: the mean temporal profiles are noisy, and their scaling does not resemble that of a single, well-defined function. Collectively, the presence of power law avalanche size and duration distributions that have exponents which satisfy the crackling relation, as well as the collapse of avalanche shapes onto a single scaling function, provides compelling evidence for criticality in cultured cortical slices.

Similar evidence of criticality *in vivo* have also been reported in the awake monkey brain [177], and in the anesthetized rat brain [178]. It has also been demonstrated that the visual cortex is self-organised to criticality by adaptation to sensory inputs [32], and more recently that whole-brain dynamics of live zebra fish [179] and mice waking from anaesthesia [180] exhibit crackling noise consistent with criticality [171]. Although there is still some

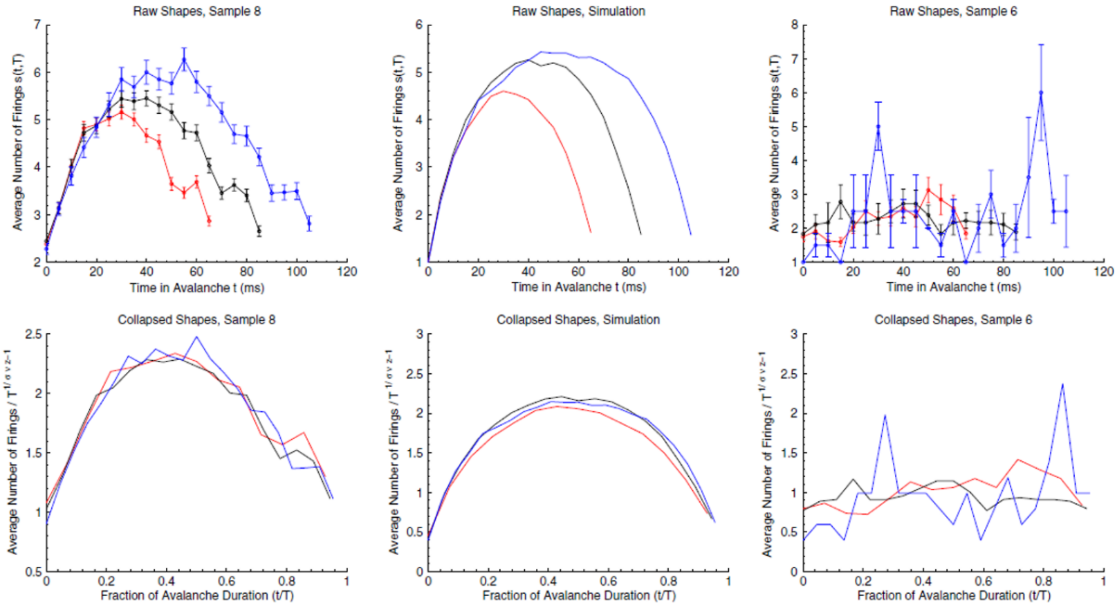


Figure 3.12: Cortical avalanches exhibit shape collapse. Shapes and attempted collapses from three data sets: two experimental and one simulated. Shapes are produced by averaging the temporal profiles of all avalanches of a particular duration; different colours here represent different durations. The collapses are plotted by rescaling the horizontal and vertical axes. The left- and right most data correspond to experimental data close to, and far from criticality, respectively. Sample 8 clearly shows the roughly parabolic shapes in the raw data and a corresponding very clean collapse, as would be expected from critical data. Sample 6 shows neither. The middle plots are a simulation of sample 8, using transfer entropy data from that set. They clearly show similar shapes and collapse to a universal scaling function. Reproduced with permission from Ref. [31].

controversy regarding criticality in the brain [181-183], the evidence in favour of criticality [184-188] is mounting steadily.

3.1.2.4 Criticality and Information Processing

Aside from explicit demonstrations of criticality, it has been shown that the brain exhibits a number of related network properties such as short characteristic path length and high clustering (small world properties) [26], hierarchical organisation [27] and scale-free degree distributions [189], and fractal structural and functional topography [141]. It is theorized that these features bestow the brain with a number of functional benefits [33], including LRTC [29,140], efficient transportation of energy and information [190], optimal dynamic range [35] and maximum number of metastable states [159], a balance between minimal wiring cost and maximum local autonomy/global connectivity [26], and robustness and adaptability leading to divergent functionality within a fixed structure [27]. Inspired by these results, neuromorphic computing approaches such as RC have begun to incorporate scale-free and hierarchical network topographies [191], which show greater performance in predictive time series tasks as compared with conventional random reservoirs [192]. The allure of the advantageous information processing features associated with criticality in neuronal networks is beginning to draw neuromorphic focus away from regular arrays of device elements towards more complex designs aimed to emulate the complexity and criticality of the brain [13], though at present, realizations of such are few and show limited functionality [66 ,193,194].

3.2 Emergent properties of PASNs

Section 3.1 introduced the concept of emergent phenomena and reviewed literature pertaining to the emergence of correlations and criticality in biological neuronal networks. It was shown that both *in vitro* and *in vivo* measurements of brain tissue are consistent with the theory of critical systems [171] where statistical analysis of avalanche dynamics are required to meet stringent criteria. These properties are thought to provide the brain with a host of functional benefits which has inspired the pursuit of neuromorphic architectures which share the emergent properties of the brain. This section presents results from electrical measurements of PASNs. The same techniques applied to biological neuronal networks are

employed to show unambiguously that PASNs exhibit LRTC and do in fact operate in a critical dynamical regime [97].

3.2.1 Self-similar Electrical Signals

The contiguous fluctuations observed in the activity of critical systems arise from the heterogeneous response of the system to slow constant driving. Thus, to investigate the presence of avalanches, DC signals were applied to two-terminal PASN devices while simultaneously measuring the conductance of the entire network (see Section 2.2.2 for details on electrical measurements). Changes in the network conductance ΔG were identified as events by applying a threshold function (Section 2.3.1) which results in a sequence of event sizes $\Delta G(t)$ as shown in the top panel of Figure 3.13 (a). The sequence is comprised of bursts

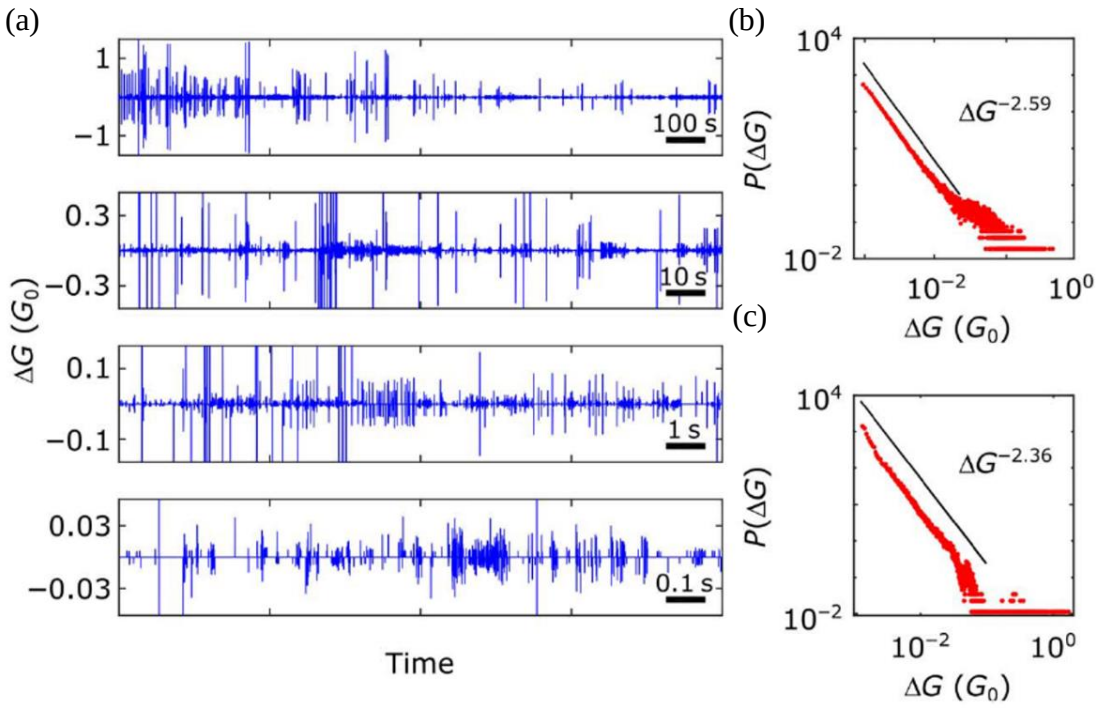


Figure 3.13: Self-similar electrical signals from PASNs. (a) Percolating devices produce complex patterns of switching events that are self-similar in nature. The top panel contains 2400 s of data, with the bottom panels showing segments of the data with 10, 100, and 1000 times greater temporal magnification and with 3, 9, and 27 times greater magnification on the vertical scale (units of $G_0 = 2e^2/h$, the quantum of conductance, are used for convenience). The activity patterns appear qualitatively similar on multiple different time scales. (b and c) The probability density function (PDF) for changes in total network conductance, $P(\Delta G)$, resulting from switching activity exhibits heavy-tailed probability distributions. The data shown in (b) (sample I) were obtained with a slow (5 Hz) sampling rate, and the data shown in (c) (sample II) were measured 1000 times faster (see Section 2.2.2), providing further evidence for self-similarity. [97]

of events of multiple sizes and are qualitatively similar to electrical signals measured in the cortex [30,31,179]. The subsequent panels show the same data magnified by 10, 100, and 1000 times on the temporal scale and 3, 9, and 27 times on the vertical scale. The burstiness of the sequence appears qualitatively similar regardless of the level of magnification, providing preliminary evidence of self-similarity.

Figure 3.13 (b, c) contain distributions of ΔG from two PASNs measured on vastly different time scales (5 Hz and 5 kHz respectively). The distributions are heavy-tailed and span about three orders of magnitude. Despite a difference in sampling rates by a factor of 1000, the slope of the linear regions is strikingly similar, estimated to be -2.59 and -2.36 respectively. This provides another promising sign of temporal self-similarity. The different event sizes correspond to different atomic switch locations throughout the highly branched network: because the local conductance of each atomic filament is nominally $1G_0$ [19], the distributions in Figure 3.13 (b, c) can only arise from a complex network of switching elements. Higher order branches of the network have more current pathways which run in parallel to them. Hence, an atomic switch on a higher order branch makes a smaller contribution to the total network conductance, and results in a smaller change in network conductance when that element switches locally from $1G_0$ to $\sim 0G_0$ (or vice versa). Conversely, low order branches which have few parallel current pathways carry more current and hence cause a larger change in the network conductance for a local conductance change which is nominally the same as that of an atomic switch on a higher order branch. To obtain the quasi-continuous ΔG distribution shown in Figure 3.13 (b, c) requires that there are a vast number of atomic switches distributed throughout a highly branched network. These observations are consistent with the spatial self-similarity found in percolating structures [195] and are evidence of fractal topology in PASNs.

3.2.2 IEI Distributions and ACFs

To investigate the presence of correlations in PASN switching activity, the $\Delta G(t)$ sequences were converted to event trains and the IEI distributions and ACFs were calculated (Section 3.1.1). Figure 3.14 (a) and (c) show the probability distributions (PDFs) of IEIs from (a) sample I (5 Hz measurement) and (c) sample II (5kHz measurement). Both follow power law decay over several orders of magnitude, consistent with Eq. (30), indicating the possibility of correlation between events. Power law exponents were found by MLE (Section 2.3.4.2) to

be $\gamma = 1.39 \pm 0.01$ and $\gamma = 1.30 \pm 0.01$ respectively. The agreement of the two exponents despite the fact that the measurement rates differ by 3 orders of magnitude is strong evidence for temporal self-similarity.

The corresponding ACFs are represented by the red curves in Figure 3.14 (b) and (d) (sample I, sample II respectively). They also follow power law decay over several orders of magnitude, indicative of LRTC. The characteristic exponent β was found by linear regression to be $\beta = -0.19 \pm 0.01$ and $\beta = -0.23 \pm 0.01$ respectively. The IEI sequences were shuffled to destroy any correlations, and the ACFs of the shuffled sequences were calculated as shown in grey in Figure 3.14 (b) and (d). The ACFs of the shuffled sequences decay much more quickly than the unshuffled sequences ($\beta = -0.66 \pm 0.01$ and $\beta = -0.64 \pm 0.02$), signalling that the original sequences indeed exhibit LRTC. Again, the fitted exponents are strikingly similar across the different devices irrespective of the different measurement rates.

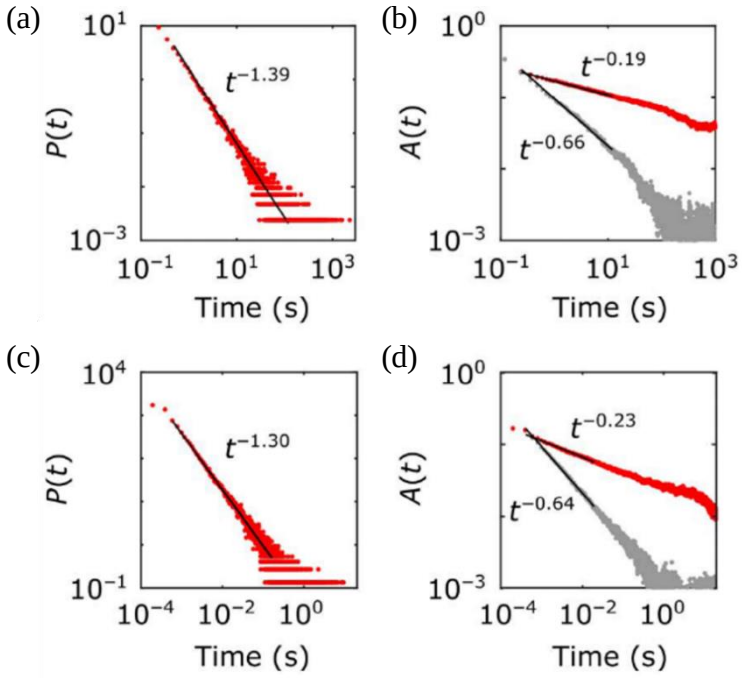


Figure 3.14: IEI distributions and ACFs from PASNs. Probability distributions of IEIs from (a) sample I (5 Hz measurement) and (c) sample II (5kHz measurement) follow power law decay over several orders of magnitude indicating the possibility of correlation between events. They have almost identical slopes despite the fact that the measurement rates differ by 3 orders of magnitude, evidence of temporal self-similarity. The corresponding ACFs (red in b, d respectively) also follow power law decay over several orders of magnitude, indicative of LRTC. When the IEI sequence is shuffled, the correlations are destroyed and the ACFs (shown in grey) decay with a much steeper slope. Again, the temporal self-similarity is evident by the near equality between the fitted exponents. [97]

The IEI distributions and ACFs of these two devices together cover more than five orders in time which is convincing evidence of scale-free temporal dynamics. Although, as discussed in Section 3.1, IEI distributions and ACFs have been shown to be insufficient methods of demonstrating that a system operates near a critical point. Thus, Section 3.2.3 presents the results from detailed avalanche analysis of PASN electrical signals.

3.2.3 Avalanche Dynamics and Criticality

Section 3.1.2 discussed the rigorous criteria for demonstrating that a dynamical system is operating in the critical regime [171]. Not only should the bursts of activity (i.e. avalanches) have power law distributed sizes and durations, but the average size of an avalanche should also scale as a power law as a function of its duration. Furthermore, the mean temporal profiles of avalanches should collapse onto a universal scaling function and estimates of the critical exponent $1/\sigma\nu z$ should be in agreement and satisfy Eq. (41). This section presents results that demonstrate that sequences of switching activity in PASNs indeed meet all of these criteria.

Avalanches were identified from event trains of PASN switching activity in accordance with the methods used in Ref. [31,171]. The mean IEI was used as the bin size Δt for each data set (the effect of varying Δt is discussed in Section 3.2.3.5). The analysis presented here was carried out using components of the MATLAB Neural Criticality and Complexity toolbox developed in Ref. [196].

3.2.3.1 Avalanche Size and Duration

Figure 3.15 shows the probability distributions (PDF) of avalanche size S from (a) sample I (5 Hz measurement) and (c) sample II (5kHz measurement). Both follow power law decay over $\sim$3 orders of magnitude, consistent with Eq. (32). The red markers show the avalanche size distributions in linear bins, which have a minimum probability between 10^{-3} and 10^{-4} corresponding to a single occurrence within the measured avalanche sequence. The same distributions are shown in logarithmic bins (blue markers) in order to demonstrate that the rare large avalanches do in fact continue to scale with the power law represented by the solid black line.

Figure 3.15 (b), and (d) show the corresponding probability distributions of avalanche duration T, which follow power law decay over $\sim$2 orders of magnitude, consistent with Eq.

(33). Again, the linearly binned data (red) appears scattered at large T, and so the logarithmically binned distributions in blue are used to demonstrate that even the longest measured avalanches conform to the fitted power law represented by the solid black line.
To verify the correlations implied by the avalanche size and duration scaling, the original IEI sequences were shuffled and the avalanche analysis repeated, producing the distributions shown in grey in Figure 3.15. All of these distributions follow the expected exponential decay consistent with uncorrelated sequences and the findings in Ref. [122].

Interestingly, the distributions of avalanche size and duration derived from the unshuffled event sequences do not show an exponential cut-off as predicted by Ref. [156]. This may be a sign that even the largest measured avalanches do not feel the finite size of the networks [156]: the cut-off is due to avalanches which propagate to the edge of the network, where it

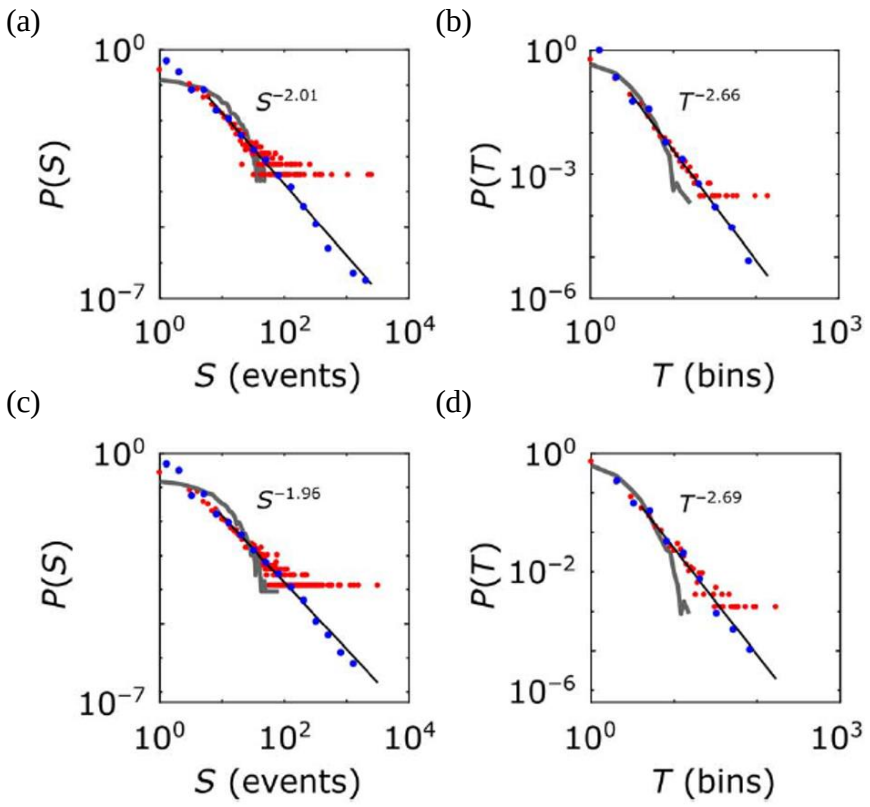


Figure 3.15: Avalanche size and duration distributions from PASN measurements. (a) The distributions shown in (a, b, sample I, 5 Hz sampling rate) and (c, d, sample II, 5kHz sampling rate) demonstrate the existence of self-similar avalanches across multiple time scales. (a, c) PDFs of avalanche size and (b, d) PDFs of avalanche duration both follow power laws with exponents $\tau \approx 2$ and $\alpha \approx 2.7$ respectively, providing strong evidence for temporal correlations. Red, distributions presented using standard (linear) bin sizes; blue, distributions presented using logarithmic bin sizes to allow visualization of the heavy tail; grey, distributions after shuffling of the sequence of IEIs in the original data to destroy correlations. [97]

may otherwise continue. The rectangular PASNs (100 × 300 μm) have hard boundaries at the electrode edges, but are not bounded at the other two edges, where the network of nanoparticles extends a large distance. As the avalanches are driven by the voltage difference between the two electrodes, switching activity should generally occur in the "active area" between them, but avalanches which propagate to the unbounded edge of the active area may continue to propagate under driving from a fringe electric field. Another possible explanation for the absence of an exponential cut-off is that avalanches in PASNs are not limited to nearest neighbour interactions as in theoretical treatments [156,171]. For example, in the Ising model of ferromagnetism [197], an event (domain flip) may cause any neighbouring domain to flip which can cause an avalanche to propagate away from the seed event. In a PASN however, when an event occurs as an atomic switch either opening or closing, the current and voltage can be redistributed throughout the entire network on a virtually instantaneous time scale. This means that when a seed event occurs in one location, the resulting instability induced which causes the next event may be in a completely different part of the network. In this explanation, avalanches need not propagate as a front of nearest neighbour induced events, and hence would not be obstructed by the edges of the network.

3.2.3.2 Average Size given Duration

The existence of power law distributed avalanche sizes and durations are promising signs that PASNs may be operating within a critical regime. However, as it has been shown that such scaling behaviour can arise away from criticality (Section 3.1.2.3) this section demonstrates that the average size of an avalanche scales with its duration as a power law, in accordance with Eq. (36).

Figure 3.16 shows the average size given duration $\langle S \rangle(T)$ for (a) sample I (5 Hz measurement) and (b) sample II (5kHz measurement). There is clear power law scaling consistent with Eq. (36), as indicated by the black curves. The exponents, which represent the critical exponent $1/\sigma\nu z$, were found (using maximum likelihood estimation (Section 2.3.4.2)) to be 1.55 ± 0.06 and 1.64 ± 0.03 for Figure 3.16 (a) and (b) respectively. Again, there is evidence of temporal self-similarity in the near equality between the exponents of the two different data sets despite being measured at different sampling rates. This type of scaling behaviour relates the spatial and temporal domains of the PASN switching activity and provides strong support for the hypothesis that such is a product of criticality. The fitting range is limited by statistical factors and does not signify limits to which the data follows a

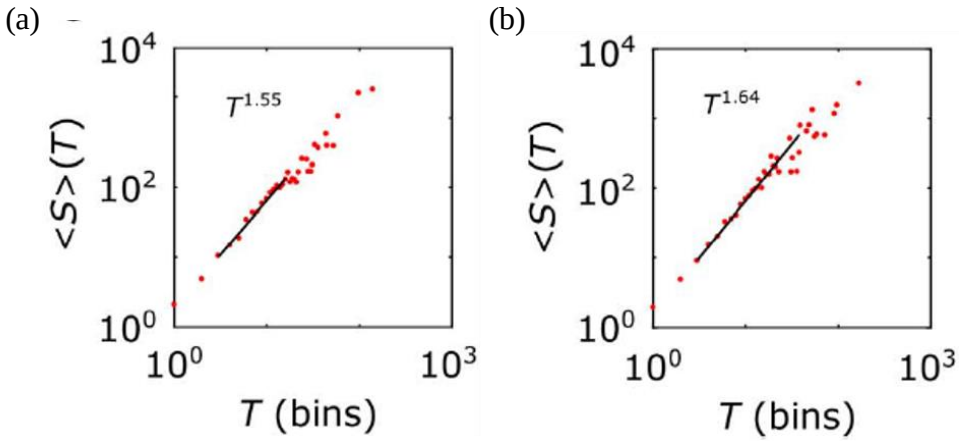


Figure 3.16: Average avalanche size given duration from PASN switching activity. Histograms of the average size $\langle S \rangle$ of an avalanche of duration (T) scales as a power law as a function of its duration for both (a) sample I (5 Hz measurement) and (b) sample II (5kHz measurement). The black lines represent the power law given by Eq. (36) with exponents found by maximum likelihood estimation. The exponents are very similar despite the different measurement rates, indicating temporal self-similarity. [97]

power law: avalanches of duration T are required to occur at least five times during the measurement to be considered statistically significant, though even those avalanches which lie outside the fitting range appear to be consistent with the scaling relationship. The lower bound represents the requirement that avalanches of $T < 3$ be excluded from avalanche shape collapse (Section 3.2.3.3) as their temporal profiles are at most two points, giving a straight line which cannot collapse onto a parabolic scaling function. As the critical exponents derived from the average size given duration and the avalanche shape collapse will be used for comparison, it is consistent to exclude avalanches of duration $T < 3$ from this analysis as well; although again it appears that these points do in fact conform to the same power laws represented by the black lines.

3.2.3.3 Avalanche Shape Collapse

The theory of critical systems predicts that the shapes of avalanches produced by critical systems should exhibit a characteristic collapse onto a universal scaling function [171]. The avalanche profiles were averaged over all occurrences of each duration by calculating the mean number of events at each time bin. This results in a single mean temporal profile for each unique duration, which are shown in Figure 3.17 for (a) sample I (5 Hz measurement) and (c) sample II (5kHz measurement). As sample I was measured on a relatively slow time scale, very long avalanches are rare and consequently there are a there are a limited number of mean temporal profiles. This is reflected by the short fitting range in Figure 3.16 (a), as

avalanches of duration T are required to occur at least five times during the measurement in order to be considered statistically significant and to be included in the avalanche shape collapse.

There exists a hierarchy amongst the avalanche profiles (best illustrated by Figure 3.17 (c)): in general, the longer an avalanche the higher its amplitude. It is exactly this characteristic which allows the mean temporal profiles to collapse onto the scaling function, as shown in Figure 3.17 (b) and (d). The shape collapse is performed by first interpolating all of the mean temporal profiles so that they have the same number of points as the longest avalanche, and then scaling time by $1/T$ (for each mean profile) so that all of the avalanche profiles span the range [0, 1]. Then, as per Eq. (40), the amplitude is scaled by T^b where the value of b is found by minimising the variance between the ensemble of temporally scaled mean avalanche profiles. The scaling function $\boldsymbol{s}(t,T)$ is then a parabola fitted using linear regression, essentially minimising the variance of the scaled collapsed profiles. The scaling

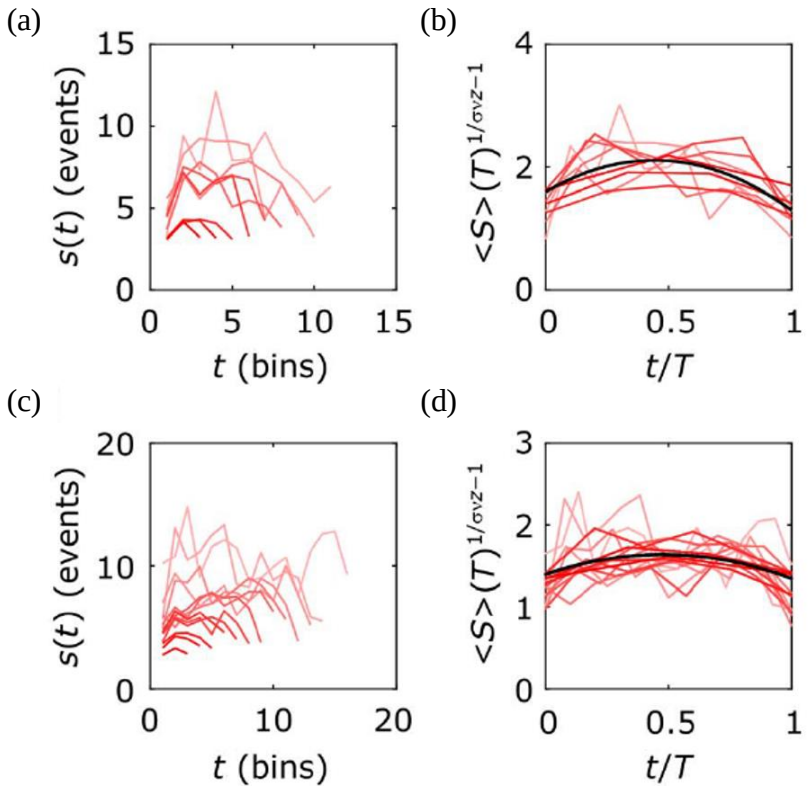


Figure 3.17: PASN avalanche shapes collapse onto a universal scaling function. The mean temporal profiles $s(t)$ for avalanches measured in (a) sample I (5 Hz measurement) and (c) sample II (5kHz measurement). The line colour fades with increasing duration to represent the decreasing number of avalanche profiles used to produce each mean profile. (b, d) When subjected to a scaling operation (see text) the mean temporal profiles from (a, c respectively) collapse onto a universal scaling function. [97]

functions are shown by the black lines in Figure 3.17 (b) and (d). Further details regarding the shape collapse methodology can be found in Ref. [196].

Because $b = 1/\sigma\nu z - 1$ (see Section 3.1.2.3), the shape collapse yields an additional estimate of the critical exponent, which in this case was found to be 1.58 ± 0.12 and 1.66 ± 0.03 for sample I and sample II respectively. As with the other exponent values found in this analysis, the agreement between the two estimates of $1/\sigma\nu z$ from the different samples and different measurement rates is striking. The shape collapses are convincing, and are consistent with the examples from neural tissue shown in Refs. [196,198].

As per Ref. [171], the observation of avalanche shape collapse is a much more robust analysis for demonstrating criticality as compared with simple power law scaling. However, as elucidated by Ref. [175], even the shape collapse can arise in systems away from criticality, and to be certain that a system is critical, one must examine the relationship between the various critical exponents.

3.2.3.4 Exponent Relationships

In accordance with Ref. [171], three independent estimates of the critical exponent $1/\sigma\nu z$ have been obtained from: the crackling relationship (Eq. (41) which considers τ and α), the $\langle S \rangle(T)$ scaling (Eq. (36)), and from the shape collapse (Eq. (40)). If the bursts of switching activity in PASNs is truly the result of criticality, then the three estimates of $1/\sigma\nu z$ should agree. The values of the critical exponents determined in this investigation are summarized in Figure 3.18: (a) and (b) show the determined values of $1/\sigma\nu z$ in green, blue and cyan (left axis), and τ and α (red, amber, right axis) for sample I and sample II respectively. The result corresponding to the data presented in Figures 3.15 - 3.17 is labelled as 'All' in both panels, representing the concatenation of the constituent datasets. The agreement between the three values of $1/\sigma\nu z$ is clear and unambiguous in both cases.

Sample I was measured at 4V, 5V, and 6V to explore the role of applied electric field strength in the avalanche dynamics. The 4V and 6V data give estimates of $1/\sigma\nu z$ which are in agreement, while the 5V dataset does not satisfy the crackling relationship (Eq. (41)). The statistics of the individual datasets are much lower than when combined, so the discrepancy between exponent values may simply arise from a statistical variation. The low statistics are not surprising given that the voltage threshold for switching is ~3V: 4-6 V is considered a low operating voltage for PASNs and the resulting switching activity is not as rapid as at

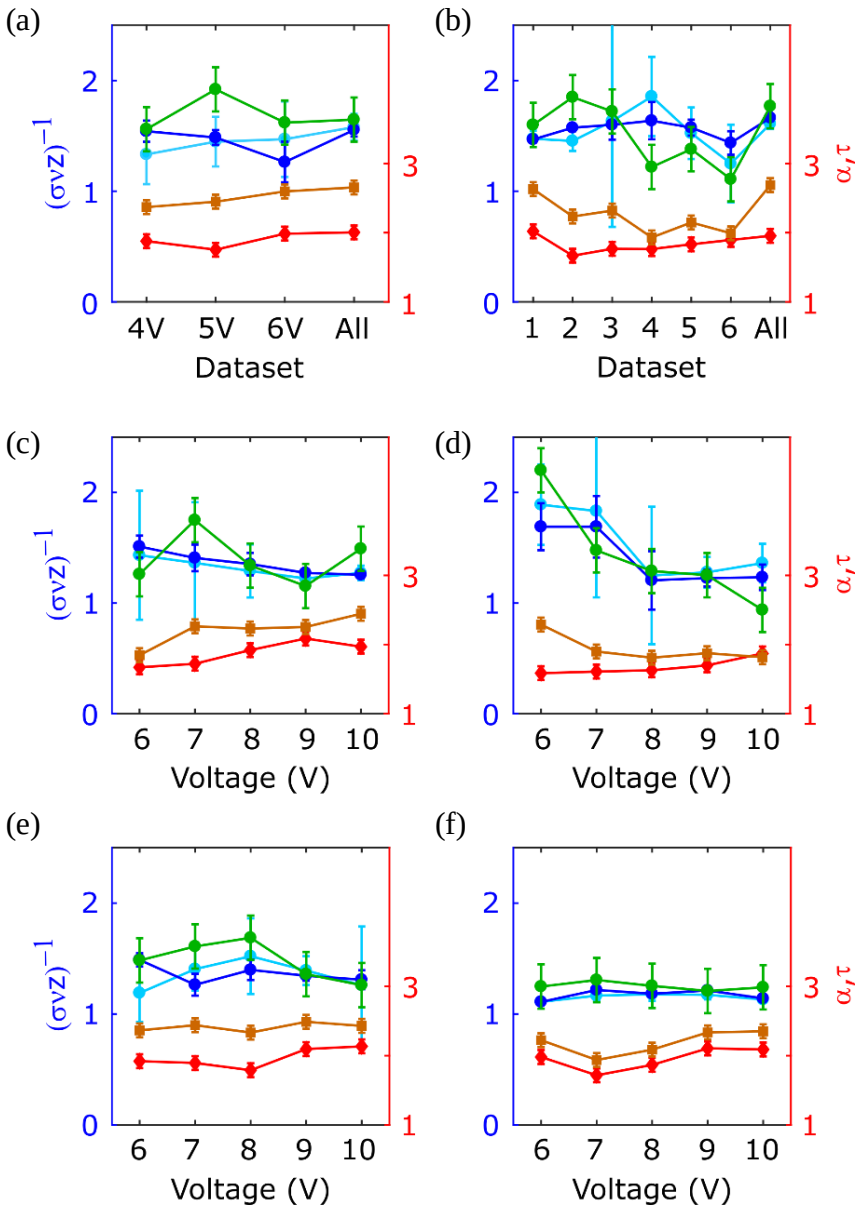


Figure 3.18: Critical exponent values from avalanches in PASN switching activity. Estimates of three independent measures of $1/\sigma\nu z$ are obtained from Eq. (41) (green), plots of mean avalanche size given duration (blue), and avalanche shape collapse (cyan). The blue, cyan, and green symbols agree within the measurement uncertainty in almost every case. Each panel also shows the power law exponents of avalanche size (τ; red) and avalanche duration (α; amber). (a) Critical exponents for sample I over a range of low voltages and for a combined dataset (5Hz sampling rate), showing that the critical exponents are substantially independent of voltage. (B) Comparison of critical exponents measured for sample II for repeated, independent 6 V DC measurements (5 kHz sampling rate). (c) Critical exponents for sample III as a function of voltage (5 Hz sampling rate). (d) Data from a second sequence of measurements on sample III identical to that in (c), showing that while the exponents α and τ vary because of internal reconfigurations of the percolating device, the three estimates of $1/\sigma\nu z$ remain in good agreement at every voltage. (e) and (f) Voltage-dependent data from sample IV (5 kHz and 5 Hz sampling rates, respectively), showing that criticality and self-similarity are observed on vastly different time scales. [97]

higher voltages. This means that a few of the rare long IEIs (i.e. from the heavy tail in Figure 3.14 (a)) may decrease the statistical quality of the data by chance. Another possibility is that the PASN system is on a chaotic phase space trajectory which oscillates about a critical point, and that the subset of that trajectory which corresponds to the 5V measurement was slightly away from the critical point. Finally, it may be possible that the system dynamics are critical at 4V and 6V, but not critical at 5V. However, as shown below, further investigations into the voltage dependence of avalanche dynamics do not support this conjecture.

Figure 3.18 (b) contains the exponent values from measurements on sample II. These recordings were made using a digital oscilloscope which had a limited sample memory of 2M points. Thus, in order to collect enough data to carry out a statistically meaningful analysis of avalanche dynamics, six measurements were made in succession and concatenated prior to the avalanche analysis. This result is labelled as 'All', while the other points along the x-axis of Figure 3.18 (b) show the exponents derived from performing the avalanche analysis on the constituent datasets. Of the six constituent datasets, the three critical exponent estimates are in agreement for all but one, with some agreeing better than others. Given that these measurements are nominally the same, it is likely that the fluctuations seen in the exponent values arise due to statistical variations. Whether this is in turn caused by a chaotic phase space trajectory along which the system moves around in the vicinity of a critical point is unclear but may be answered by computer simulations in some future work. The critical exponents from the 'All' case are in very good agreement, indicating that discrepancies in the shorter datasets are likely due to statistical limitations.

A range of measurements were also carried out on two additional samples. Figure 3.18 (c) and (d) show the resulting critical exponents for different DC voltage measurements on sample III, all measured at 5kHz. There is only one out of the ten datasets which produce $1/\sigma\nu z$ values which do not agree: the crackling relationship estimate (green) for the second 6V measurement is not in agreement with the $\langle S \rangle(T)$ estimate (blue). The critical exponents of the all of the other datasets are consistent with criticality, despite the range of different voltages (between 6V and 10V). This demonstrates that the observation of criticality does not depend strongly on the driving strength, consistent with SOC. For example, the avalanches in a sand pile would not depend on how fast individual grains are added to the pile, until the rate of addition became comparable to the time scale of the longest avalanches; One grain per day, or per hour, or per minute would make no difference, however, 1 grain

per μs would likely affect the avalanche dynamics, probably driving it into a supercritical state where one single continuous avalanche spans the entire system.

Repeated measurements of sample III at the same voltage give critical exponent values which differ slightly, exemplified by the fluctuations in Figure 3.18 (c) and (d), and the differences between the two panels. Again, these measurements are nominally the same, the only difference being the initial configuration of the PASN for each measurement. The state of the network at any given time is the result of all previous stimulation and the switching activity which it induced. Despite the fluctuations in the values of the critical exponents, they agree in almost all cases, which is suggestive of a chaotic phase space trajectory, typical of critical systems [199].

To further demonstrate the reproducibility of the results, additional voltage dependent measurements were made on another PASN, sample IV. Figure 3.18 shows the resulting

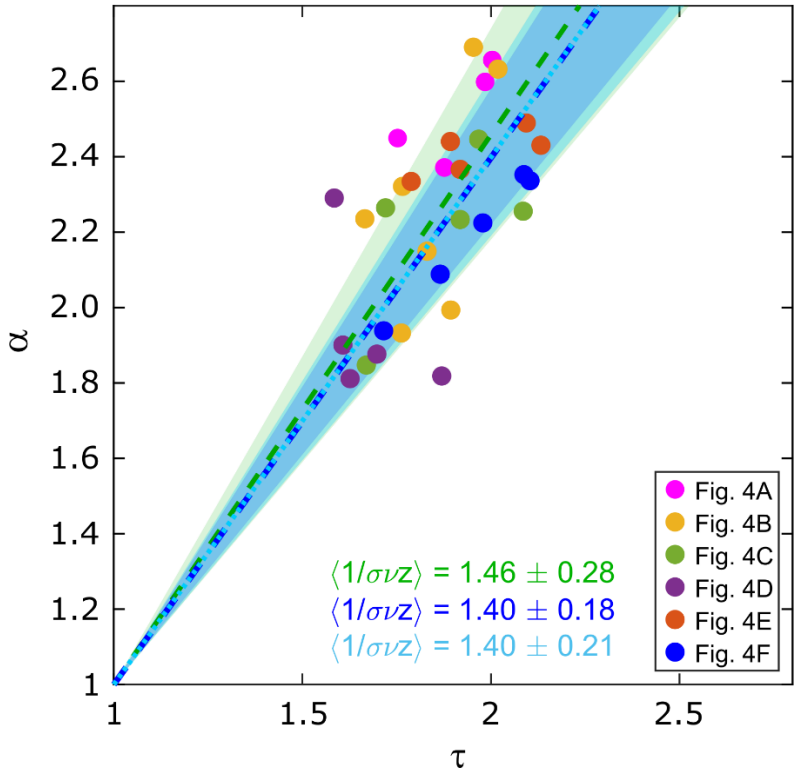


Figure 3.19: Scatter plot of the critical exponents from avalanches in PASNs. Different coloured markers represent the values of α and τ from each of the different panels in Figure 3.18. The mean values of $1/\sigma\nu z$ are expressed as the slopes of the broken lines, which were found to be 1.46 ± 0.05, 1.40 ± 0.03, and 1.40 ± 0.04 from the crackling relationship (green), $\langle S \rangle(T)$ (blue), and shape collapse (cyan) respectively (uncertainties given by shaded segments are 1 standard deviation), indicating that there is no significant difference between the estimates from the three independent methods. This is confirmed by a single-factor analysis of variance (ANOVA) test ($P = 0.47$). This is strong evidence for universal avalanche dynamics in PASN devices. [97]

critical exponent values for (e) 5 kHz sampling rate, and (f) 5 Hz sampling rate. Even measuring the same sample at rates which differ by three orders of magnitude produces critical exponent values which are similar and in agreement, showing that the analysis is robust against the choice of sampling rate. Again, fluctuations are observed due to the chaotic trajectory of the PASN which produces statistical variations.

In Figure 3.18 the format used to show the critical exponents makes the agreement between the values of $1/\sigma\nu z$ for each measurement clear, but it is difficult to see how they compare as an ensemble. Hence, the same τ and α values (all points in red and amber in Figure 3.18) are shown as a scatter plot in Figure 3.19. The crackling relationship (Eq. (41)) expresses $1/\sigma\nu z$ in terms of τ and α, meaning that $1/\sigma\nu z$ can be expressed as the slope of a straight line with an offset of 1 along both the τ and α axes:

$$\frac{1}{\sigma\nu z} = \frac{\alpha - 1}{\tau - 1} \quad \rightarrow \quad \alpha = \frac{1}{\sigma\nu z}(\tau - 1) + 1 \tag{42}$$

The mean values of $1/\sigma\nu z$ were calculated independently for the green, blue and cyan points in Figure 3.18, and are represented by the straight lines in Figure 3.19. The colours are consistent with those used in Figure 3.18: crackling relationship (green), $\langle S \rangle(T)$ scaling (blue), and shape collapse (cyan). The mean values of $1/\sigma\nu z$ were found to be 1.46 ± 0.05, 1.40 ± 0.03, and 1.40 ± 0.04 respectively, where the uncertainties are the standard deviation of each set of exponent values, and are represented by the shaded segments in Figure 3.19. The agreement of the $\langle 1/\sigma\nu z \rangle$ values, as well as the fact that the majority of the τ and α points fall within the standard deviation, are strong indicators of universal avalanche dynamics across all of the different measurements performed in this study [200]. This was further verified by a single factor ANOVA test which gave a p-value of 0.47, signalling that there is no significant difference between the sets of $1/\sigma\nu z$ values obtained by the three independent methods.

The results presented in this chapter satisfy some of the most stringent criteria for demonstrating criticality in a dynamical system [171]. Even sceptics of the traditional analysis of criticality, which relied solely on the presence of power law scaling (See Section 3.1.2.3), suggest that the agreement of critical exponent values is unlikely to arise away from criticality and should therefore be sought for demonstrations of such [175]. The fact that DC measurements of PASN switching activity over a range of different devices, operating voltages, and measurement rates, produce critical exponents which are in agreement in almost all cases, is strong evidence that PASNs are in fact critical systems.

3.2.3.5 Choice of time bin size (Δt) and ΔG threshold

One important parameter choice which affects the analysis of avalanches and the resulting exponent values is the size of the time bin Δt. A common method [30,31] is to use the mean IEI as the bin size, a practise which was adopted and used to produce the results of this chapter. However, as this is a free parameter, it is prudent to explore its effect on the analysis.

Figure 3.20 presents the results of avalanche analysis on sample II using a wide range different sized time bins: (a) shows the distributions of avalanche size, (b) shows the distributions of avalanche duration, and (c) show the resulting critical exponent values for each bin size in the range 0.001 – 1 s. As the bin size increases, small avalanches are concatenated into larger avalanches due to the fact that avalanches must be separated by at least one inactive time bin. This results in an increasing proportion of large avalanches and causes the slopes of the size and duration distributions to decrease. This is reflected by the decreasing values of τ and α as a function of increasing bin size in Figure 3.20 (c). Remarkably, while τ and α decrease significantly, and the size and duration distributions in Figure 3.20 (a) and (b) change in terms of their slope and total number of avalanches, all but one bin size ($\Delta t = 0.1$ s) result in values of $1/\sigma\nu z$ which are in agreement. This demonstrates

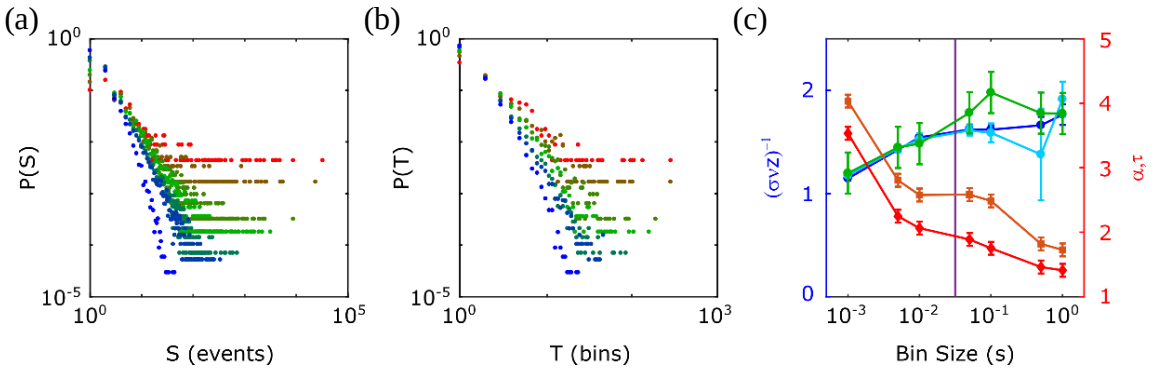


Figure 3.20: Avalanche analysis using different sized time bins. The detection of avalanches requires events to be binned in the time domain (Section 3.1.2). The size of this time bin was chosen in each case to be the mean IEI, and here it is shown that the conclusions of this chapter do not rely on this choice. (a) The avalanche size distributions, and (a) the avalanche duration distributions, for different bin sizes (0.001 – 1 s, blue - red). (C) The power law exponents of the size and duration distributions do change as a function of bin size (red and amber), but they do so in such a way that the three estimates of $1/\sigma\nu z$ (crackling relationship: green; <S>(T): blue; shape collapse: cyan) are still consistent with criticality: green, blue, and cyan symbols agree within the uncertainties for almost every bin size. Note that the uncertainties increase for large bin sizes because the number of avalanches decreases significantly. The mean IEI is indicated by the vertical purple line. [97]

that the choice of Δt does not significantly affect the results of the avalanche analysis and provides further evidence for temporal self-similarity in PASN switching activity.

Another free parameter which may bias the results of the avalanche analysis is the choice of ΔG threshold which is used to identify switching events from the PASN conductance measurements. All electrical measurements are subject to some level of noise which appears as random fluctuations. As demonstrated in Figure 3.13, the distribution of event sizes ΔG is approximately power law with many more small events than large events. The smallest measurable changes in conductance are on the order of the noise amplitude and so a distinction must be made between those fluctuations which arise due to switching events and those which correspond to noise. The threshold is therefore set just above the noise amplitude to be certain that the detected events are in fact due to switching. Further details on event detection can be found in Section 2.3.1.

The threshold used in this study was 0.005 G_0, which is above the noise level. Because the sizes of events likely extend below the noise amplitude, there are always going to be some missed events, and the aim of the thresholding procedure is to maximize the number of captured events while ensuring that no false events are recorded. There is therefore some ambiguity about where the threshold should be set, and here it is shown that using a smaller

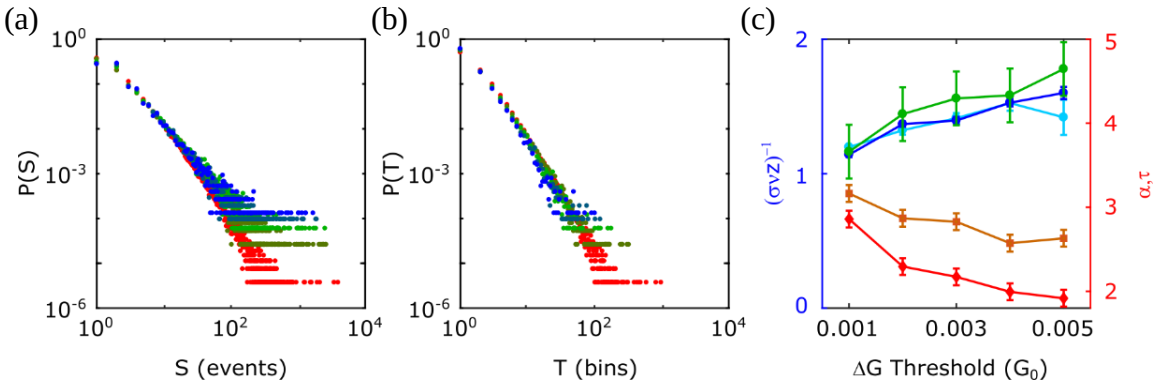


Figure 3.21: Avalanche analysis after using different event detection thresholds. The value of the threshold is chosen to be close to but above the noise level. The data presented in this figure is from sample II but data from all samples show similar behaviour. (a) The avalanche size distribution is not significantly affected by changes to the threshold value over the range 0.001-0.005 G_0 (red - blue). (b) The avalanche duration distribution is similarly robust against changes in the threshold value. (c) Although there is some variation in the values of the fitted exponents (τ and α) as a function of the threshold value (due mainly to the change in mean IEI value that is used for binning the data), the value of $1/\sigma\nu z$ derived from the crackling relationship (green) is in good agreement with the $1/\sigma\nu z$ values obtained from the $\langle S\rangle(T)$ (blue) and shape collapse (cyan) over the entire range of threshold values. The demonstration of criticality is therefore not affected by the event detection procedure. [97]

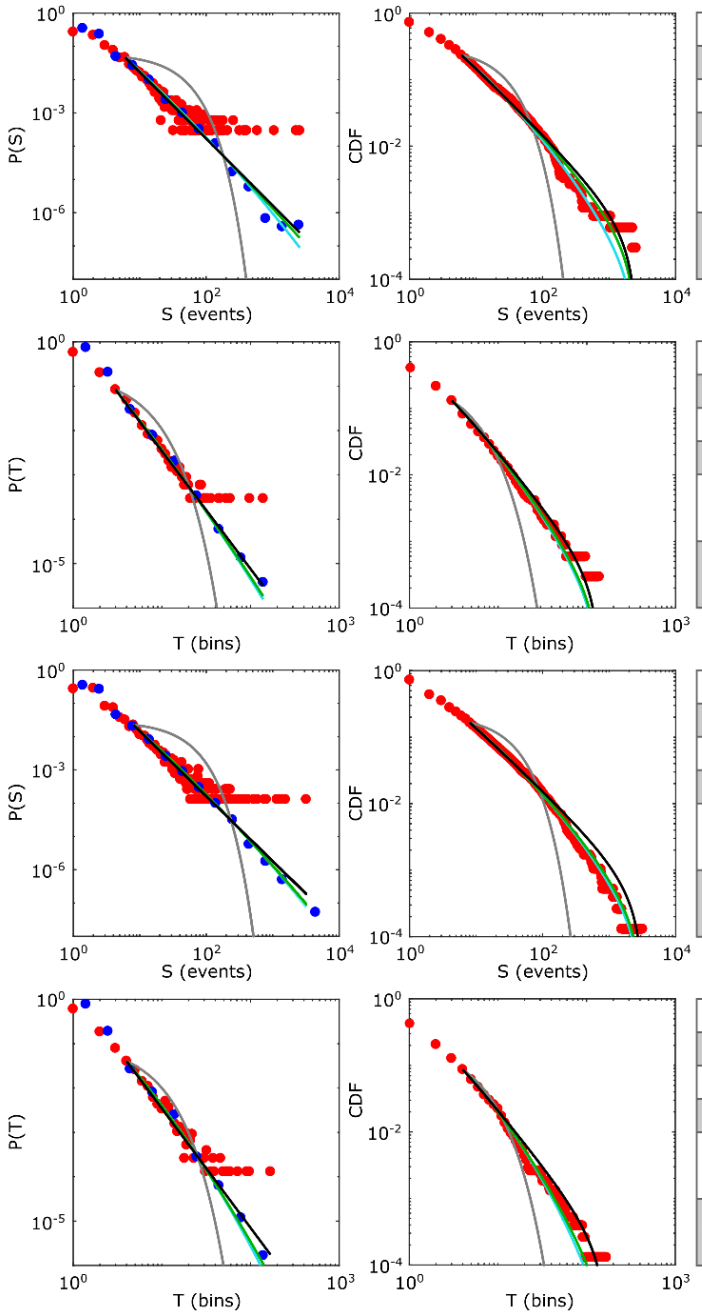
threshold within a reasonable range does not significantly affect the results of the avalanche analysis.

Figure 3.21 contains the results of avalanche analysis performed on sample II after using a range of different thresholds during the event detection procedure: (a) shows the avalanche size distributions, (b) shows the avalanche duration distributions, and (c) shows the values of the critical exponents as a function of ΔG threshold over a range 0.001-0.005 G_0. Below 0.001 G_0, the detection procedure is definitely picking up noise, so the exploration of threshold effects is limited to that value. As the threshold increases, the number of detected events decreases leading to longer IEIs and a longer $\langle IEI \rangle$ (which is used as the time bin size). Hence, increasing the ΔG threshold has a similar effect to increasing the time bin size Δt (discussed above): the size and duration distributions become slightly flatter, leading to a small decrease in the values of the critical exponents τ and α, as shown in Figure 3.21 (c). Despite this, the values of $1/\sigma\nu z$ are in good agreement at every threshold tested, demonstrating that the choice of ΔG threshold is substantially independent of the conclusion that PASNs operate in a critical regime.

3.2.3.6 Statistical Verification of Power Law Distributions

The avalanche analysis presented in this chapter is a more complete and robust demonstration of criticality than traditional methods which relied solely on power law scaling. Nonetheless, this analysis still hinges heavily on the presence of power law scaling and on the fitted exponents. Therefore, several statistical tests were conducted to be certain that the avalanche size and duration distributions are indeed power law and that the fitted exponents are correct.

All of the histograms and probability distributions presented in this chapter were tested against power law, exponential, log-normal, and Weibull functions using the BIC (Section 2.3.4.4). Figure 3.22 presents the results of this verification process. The avalanche size and duration PDFs (left) from sample I and sample II (as in Figure 3.15) are accompanied by their corresponding CDFs (middle). In each panel, all four functions being tested are shown in different colours: power law ($x^{-\alpha}$, black), exponential ($e^{-\lambda x}$), gray), log-normal ($x^{-1}e^{-\sigma^{-2}(ln\,x-\mu)^2/2}$, green), and Weibull ($x^{\eta-1}\,e^{-\gamma x^{\eta}}$, cyan). The tables to the right contain the parameters of each model which were fitted using MLE (Section 2.3.4.2). The different functions were assessed using the BIC, the results of which are characterised by w_{BIC} which



Avalanche Size - Sample I			
Distribution	Parameters	ln(L)	w_{BIC}
Power law	τ = 2.01	-3517.5	0.93
Exponential	λ = 0.04	-4076.8	0.00
Log-normal	μ = -18.80 σ^2 = 22.16	-3517.0	0.05
Weibull	η = 0.10 γ = 7.77	-3517.9	0.02

Avalanche Duration - Sample I			
Distribution	Parameters	ln(L)	w_{BIC}
Power law	α = 2.66	-1494.5	0.89
Exponential	λ = 0.29	-1630.6	0.00
Log-normal	μ = -9.21 σ^2 = 6.80	-1494.0	0.06
Weibull	η = 0.10 γ = 14.26	-1494.1	0.05

Avalanche Size - Sample II			
Distribution	Parameters	ln(L)	w_{BIC}
Power law	τ = 1.96	-5881.8	0.45
Exponential	λ = 0.03	-6578.3	0.00
Log-normal	μ = -6.51 σ^2 = 10.80	-5878.7	0.25
Weibull	η = 0.10 γ = 7.17	-5878.6	0.30

Avalanche Duration - Sample II			
Distribution	Parameters	ln(L)	w_{BIC}
Power law	α = 2.69	-2278.2	0.47
Exponential	λ = 0.23	-2407.8	0.00
Log-normal	μ = -2.80 σ^2 = 3.03	-2275.1	0.31
Weibull	η = 0.20 γ = 5.89	-2275.5	0.22

Figure 3.22: Comparison of power law and other fits to the avalanche size and duration distributions. Left: probability density functions, reproduced from Figure 3.15; centre: corresponding complementary cumulative distribution functions (CDF); right: tables of fitted parameters. Fitted curves are obtained for the data over the same ranges as are shown in Figure 3.15 using MLE. The fits are compared using the Bayesian information criterion (BIC). The weights (w_{BIC}) indicate the relative probability that each model provides the best fit. The compared distributions are power law ($x^{-\alpha}$, black), exponential ($e^{-\lambda x}$), gray), log-normal ($x^{-1}e^{-\sigma^{-2}(\ln x-\mu)^2/2}$, green), and Weibull ($x^{\eta-1}\ e^{-\gamma x^{\eta}}$, cyan). Note that the log-normal and Weibull fits are so similar that it is difficult to resolve the green and cyan lines. In each case the power law distribution is favoured over all the others. [97]

gives the probability that a particular function gives the best fit to the data. In every case the power law is found to be the best fit.

While the log-normal and Weibull distributions produce fits that appear to the naked eye comparable to the quality of the power law fit, the BIC takes into account the larger number of parameters required to achieve the fit. Note also that the mean of the log-normal distribution is $e^{(\mu+\sigma^2)/2}$ and the mean of the Weibull distribution is $\gamma^{-1/\eta}\Gamma(1+\eta^{-1})$ which are smaller than 1 (and in many cases smaller than 0.01) for every fit. These values are clearly unphysical and are incompatible with the actual means of the distributions. The unphysical fitted parameters are chosen by the fitting procedure because they result in log-normal and Weibull distributions that approximate power laws in their tails.

The fitted power law functions were then subjected to the KS test (Section 2.3.4.3) where it is required that $p < 0.2$ to reject the null hypothesis that the avalanche distributions are power law distributed. In all cases, it is found that this null hypothesis cannot be rejected ($p > 0.2$), leading to the conclusion that the sizes and durations of avalanches in PASNs are indeed power laws.

3.2.4 Conclusions

The aim of this chapter was to determine whether the switching activity observed in PASNs is consistent with the avalanches of criticality that exist in cortical tissue. It is hypothesized that the critical regime in which the brain operates provides many computational advantages, and that the same features may be beneficial for brain inspired computing approaches such as neuromorphic computing. Statistical analysis of the IEI probability distributions showed that they follow power law decay, indicative of correlated switching, which was supported by analysis of the ACFs which show slow power law decay corresponding to LRTC. The presence of correlations was verified by shuffling the original sequences which destroyed the correlations and produced exponential distributions which were markedly different than those of the original data.

The distributions of avalanche sizes and durations were also shown to follow power laws, the critical exponents of which were determined by MLE and verified by the KS test. Power law distributed avalanche statistics are a sign of scale-free dynamics, which is consistent with observations of neuronal firing patterns. Further to this, it was demonstrated that the average size of avalanches in PASNs scale as a power law as a function of their duration, and that the

average shapes of avalanches collapse onto a universal scaling function, providing robust evidence for criticality. Finally, it was shown that the relationship between the critical exponent values are consistent with the unified theory of critical systems, a property which is considered to be one the most stringent criteria for demonstrating that a dynamical system is operating in a critical regime.

Together, the evidence presented in this chapter makes a strong case that PASNs are critical systems and that their measured electrical signals exhibit many of the same statistical properties as the brain. This is a promising sign that PASNs may have computational power and paves the way towards the realization of a novel neuromorphic architecture that emulates the complexity and criticality of the biological brain; a characteristic which has so far been lacking from neuromorphic approaches [13].

Chapter 4

Introduction to Time-delay Reservoir Computing

Section 1.4 introduced the different modes of nonlinear conduction in PASNs. Below a threshold voltage, the PASN is in a passive tunnelling regime (Section 1.4.4) in which the network topology remains fixed and conductance depends on applied voltage. Voltages which exceed the threshold push the PASN into an active switching regime (Section 1.4.5) where filaments are created and destroyed within tunnel junctions, resulting in a dynamic network topology and a voltage dependent switching rate. Chapter 3 demonstrated that in the active regime, switching events occur in avalanches that exhibit many scale-free properties which are consistent with LRTC and criticality. It is believed that critical networks, such as those in the cortex, offer a host of computational advantages over non-critical networks, including maximum computational power, information transfer and storage [13,33-36].

This chapter discusses a computational framework called time-delay reservoir computing (TDRC) in which the properties of PASNs outlined above are utilized to attempt pattern recognition and prediction tasks. Section 4.1 Introduces TDRC as an adaptation of

conventional RC which uses a single nonlinear dynamical node (NDN) to emulate a virtual reservoir. Section 4.2 describes several benchmark tasks commonly used to demonstrate computational abilities of novel neural networks/neuromorphic systems. In Section 4.3, a complete overview of the TDRC framework is given and the pre-processing and training procedures used in this thesis are described in detail. Section 4.4 contains a review of literature regarding novel implementations of physical TDRC systems. In Chapter 5, several software implementations of TDRC are explored and used to assess different NDN models and parameter choices. Finally, in Chapter 6 an experimental PASN-based TDRC implementation is used to perform two benchmark tasks, and Section 6.3 details plans for future work in this area, as informed by the simulation and experimental results.

4.1 Introduction to TDRC

Section 1.2.3 introduced reservoir computing (RC) as an adaptation of the recurrent neural network (RNN) model (Section 1.2.2). In RC, only output connections are trained and the hidden layer, or *reservoir* in this context, remains untrained. Motivated by the laborious training process required to implement RNNs, the RC approach significantly reduces the training cost for tasks like pattern recognition, classification and time-series forecasting. For these tasks, RC has been shown to compete with, and in some cases exceed the performance of traditionally trained RNNs [201]. RNNs are software implementations of bio-inspired computing principles, hence the original implementations of RC were also in software. More recently the RC framework has been employed in an increasing number of neuromorphic applications which has enabled the utilization of nonconventional computing hardware [201-204] (discussed in Section 1.2.3).

Why should PASNs be suitable for RC?

PASNs are networks of nonlinearly interacting elements (tunnel gaps/atomic switches) which respond to external electric fields. It is therefore a reasonable conception that a PASN may be a suitable hardware platform with which to implement a neuromorphic RC system. If a PASN was used as a physical reservoir in place of the hidden layer and input signals encoded in applied voltage stimulus, then the higher-dimensional transformation (Section 4.3.4.1) which lies at the heart of neural network computation may indeed be carried out by the physical conduction mechanisms taking place within PASNs. As this transformation is the

most costly aspect of software-based RC approaches, a hardware-based approach using a PASN may yield a RC system which can perform classification/recognition/prediction tasks with superior power efficiency [51].

Why explore the time-delay variant of RC?

The output of the hidden layer of a neural network has high dimensionality because there are many nodes within the network which have connections to the output layer, as illustrated by the red arrows in Figure 4.1. Training an RC system involves adapting the output connection weights such that the signal at the output nodes approximates the desired response. In hardware-based RC systems, the output layer and training are facilitated by a separate entity than the reservoir; typically by dedicated CMOS circuitry, or by a conventional computer [204-207]. Therefore, to use a PASN as a physical reservoir would therefore require multi-contact device geometry: N electrodes which are distributed spatially throughout the network of nanoparticles, providing N different (but correlated) nonlinearly transformed representations of the input signal to the output layer. This is indeed a feasible idea, but at the time of this study, multi-contact PASNs are in their infancy and are not available for neuromorphic applications. Hence, to explore the computational capabilities of the two-contact PASNs studied in this thesis, an alternative RC framework called TDRC is used.

TDRC is a variant of RC which uses a single nonlinear dynamical node (NDN) to emulate a reservoir of nonlinearly interacting, spatially distributed nodes. Because TDRC requires only one real node, it is a suitable framework to explore computation using the two-contact PASNs studied in this thesis.

The difference between RC and TDRC

A full overview of the TDRC framework is given in Section 4.3 but the difference between RC and TDRC is summarised schematically in Figure 4.1 [202]. In RC, the connections between the nodes in the reservoir ensure that the state of each node depends on the states of the other nodes (spatial correlation), and the recurrent connections ensure that the state of each node depends on the previous state of the reservoir (temporal correlation). In TDRC, the reservoir is replaced with a delay line: a period of time during which the state of a single NDN is sampled at regular intervals, with each sample representing the state of a virtual node. If the dynamical node is operating in a transient regime (i.e. never reaching a steady state), then its state at any time is dependent on previous states, thus providing temporal

correlation between the virtual nodes: an analogue of the spatial correlation between nodes in RC (discussed further in Section 4.3.4).

Each loop around the delay line is equivalent to a time step τ, such that a virtual reservoir of N virtual nodes can be described by a state vector of length N at every time step. The temporal correlation which is provided by the recurrent connections in RC is emulated in TDRC by a delayed feedback mechanism: the state of each virtual node on the k^{th} loop of the delay line is fed back into the NDN during the $(k+1)^{th}$ loop, thus providing a dependence between the states of the virtual reservoir at each time step.

In RC, the input signal is fed simultaneously into the nodes of the reservoir via the blue connections in Figure 4.1 (a). This is impossible in TDRC because the virtual nodes are points

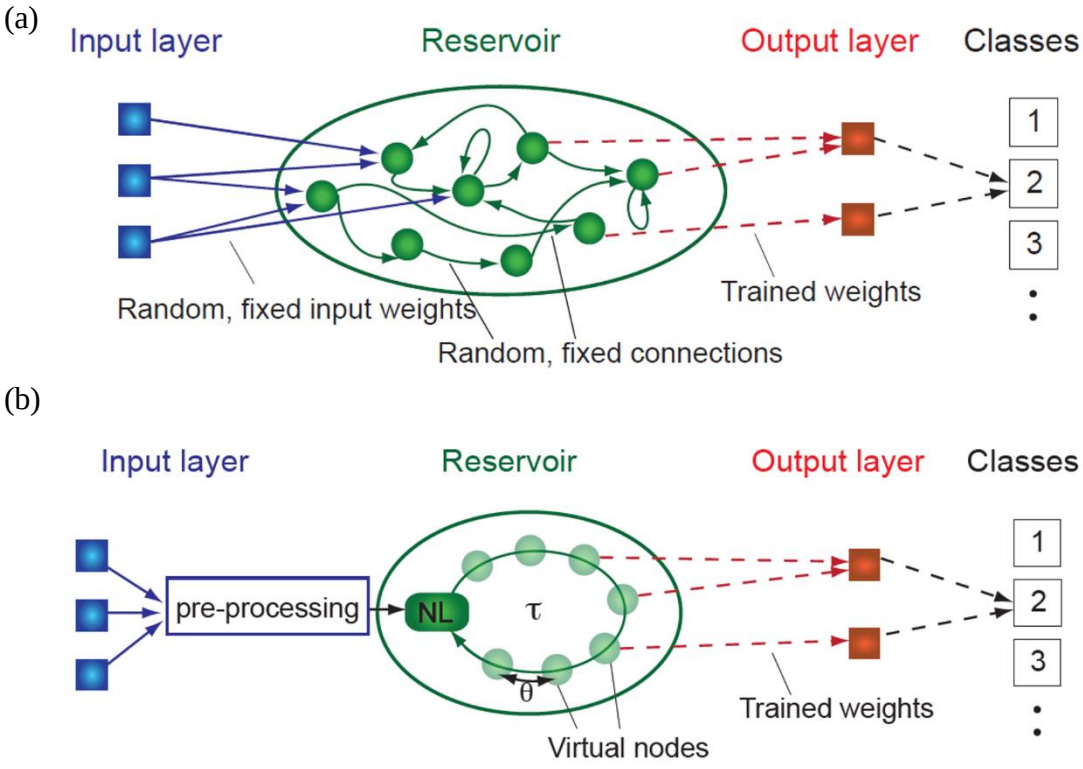


Figure 4.1: Conventional RC scheme vs TDRC scheme. (a) A schematic diagram of conventional RC. Input nodes are randomly connected to the reservoir and input signals enter the reservoir in parallel. The spatial and recurrent connections between the nodes facilitate the projection of input signals to a higher-dimensional reservoir space. The output layer is a linear read-out mechanism which is trained to map the reservoir state to the desired output by adapting the connection weights represented by the red arrows. (b) A schematic diagram of TDRC. Inputs are flattened and fed sequentially into a single nonlinear dynamical node (NDN) represented by NL. Sampling the state of the NDN at regular intervals (θ) emulates a virtual reservoir of virtual nodes. The NDN must be kept in a transient state in order to facilitate temporal connections between virtual nodes, and recurrent connectivity is provided by an external feedback mechanism (not shown). Reproduced from Ref. [202] under CCL.

in time rather than points in space, hence the input signal must be flattened in a pre-processing step (Section 4.3.1). The output layer and training schemes used in TDRC (Section 4.3.5) are identical to those used in RC: some training data is fed into the reservoir and the state of each node is sampled at each time step producing a time-dependent response vector for each of the N nodes in the reservoir. The final output of the reservoir is a linear combination of the response vectors and the training occurs by adapting the weights (i.e. the constant factors w_i in the linear combination described by Eq. (46)) so that the linear combination approximates the desired output as closely as possible.

Once the training is complete, testing of the system is performed using new data which is qualitatively similar to the training data. If the training was successful and the weights have been successfully adapted, then the reservoir will be able to generate a good approximation of the desired output, even though it has never been exposed to that input data before i.e. the system has "learned" to correctly identify patterns in the input signal.

4.2 Tasks and Benchmarking

Before moving on to a detailed description of the TDRC framework, it is useful to first introduce some of the benchmark tasks commonly used by the neuromorphic community. Conventional CMOS computer chips can all be compared quantitatively by technical specifications. For example, a CPU which has a clock speed of 1 GHz is better than a CPU operating at 1 MHz, 16 GB of RAM is better than 8 GB of RAM, etc. But neuromorphic computing is an emerging field comprising many different types of approaches and systems, which in most cases cannot be directly compared with one another in terms of technical specifications. This necessitates that neuromorphic architectures be assessed using their performance at some well-defined brain-like tasks (pattern recognition/classification/ prediction). This section describes several of the tasks commonly used to quantify the performance of reservoirs in RC.

It should be noted that only the waveform discrimination (Section 4.2.2) and NARMA10 (Section 4.2.4) tasks are explored in detail in this thesis. The remaining tasks outlined here are included to: support the review of literature results in Section 4.4; present alternative tasks which may be explored in future works in order to comprehensively demonstrate the computational abilities of PASN-based neuromorphic architectures.

4.2.1 Exclusive disjunction

Exclusive disjunction, more commonly known as exclusive OR (XOR), is a Boolean logic operation which takes two binary inputs and outputs TRUE if those inputs differ and FALSE otherwise. It is closely related to the OR logic operation, which returns TRUE if either one of its inputs are TRUE, but XOR returns false for the case where both inputs are TRUE. As a classification task, the XOR task takes pairs of binary values (representing TRUE and FALSE) as inputs and requires a binary output corresponding to TRUE if the two input values are different, and FALSE if they are the same.

As shown in Figure 1.4, XOR is a nonlinearly separable classification problem: the two classes TRUE and FALSE (denoted by the black and white circles) cannot be separated by a single linear boundary. This problem was studied extensively in the early days of neural network software [46,208] and is now commonly used only for illustrative purposes [38] and exploring the capabilities of new architectures [66].

4.2.2 Waveform Discrimination

Waveform discrimination [146] is a classification task in which the input is a randomised sequence of different waveforms (e.g. sine and square) of equal period and amplitude. The task is to correctly classify each waveform in the sequence by outputting a different binary value for each class (e.g. 1 for a sine wave and a -1 for a square wave). The task may be adapted to include n waveform types, which typically involves a "winner-takes-all" readout (Figure 4.2): n linear output nodes (i.e. perceptrons; Section 1.2.1) are trained to each return '1' for one class and zero for the others. The reservoir states are read by each output node and the one which returns the closest value to 1 is the winner: the input waveform is identified as the winning class.

The performance of the waveform discrimination task is characterised by the classification score: the proportion of waveforms which were classified correctly. The classification score is often expressed as a percentage of the total number of testing examples. The performance may also be characterised using the error between the virtual reservoir output $\hat{y}$ and the target function y. The error is often expressed as the normalised root-mean

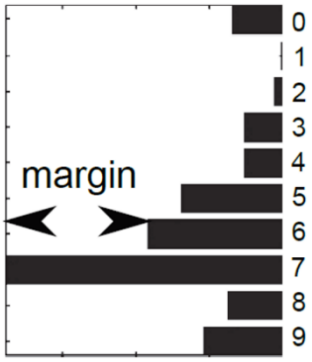


Figure 4.2: Winner-takes-all output layer. The output layer features one linear classifier for each of the classes in the input data. Each classifier is trained to output one for a particular class, and zero for all other classes. The classifier which outputs the highest value is the 'winner' and determines the class to which the input should belong. The margin is the difference between the output values of the winning classifier and the second place classifier indicates of the level of certainty of the classification.

square error (NRMSE) between the reservoir output $\hat{y}_k$ and the target function y_k:

$$NRMSE = \sqrt{\frac{1}{m}\frac{\sum_{k=1}^{m}(\hat{y}_k - y_k)^2}{\sigma^2(y_k)}} \tag{43}$$

where σ represents the standard deviation and m is the number of time steps in the target function. Better performance at the task corresponds to a lower NRMSE value.

As in the case of XOR, this type of classification task requires nonlinear separation, but it is the classification of *sequences* of input values, whereas XOR uses static inputs. Therefore, this task is particularly well suited to demonstrate the effects of memory in neural networks and has been used to assess several TDRC implementations [146,201,209,210] where temporal correlation facilitates high dimensional projection within the reservoir. In this thesis, the sine-square waveform discrimination task is used extensively to study both numerical and experimental implementations of PASN-based TDRC systems.

4.2.3 Nonlinear Channel Equalisation

The nonlinear channel equalization task is based on a real-world problem in wireless communications. A sequence of characters $s(n)$ to be sent wirelessly to a receiver is first converted to an analogue envelope signal $d(n)$ and then modulated on a high-frequency carrier signal. The receiver demodulates the carrier to produce an analogue signal $u(n)$, which is a corrupted version of $d(n)$. The corruption is caused by thermal/interference noise, nonlinear distortion through high-gain amplifiers, and multipath propagation causing inter-

symbol superposition [192]. The task is to de-corrupt the signal $u(n)$ so that it resembles $d(n)$ as closely as possible. The de-corrupted signal is then converted back into characters, which should be the same as the original input sequence $s(n)$. The performance metric for this task is the error rate: the fraction of reconstructed characters which do not match those in $s(n)$.

This task has been used to demonstrate practical utility for neuromorphic architectures [201,211], while many of the other common tasks (e.g. waveform discrimination) are used solely for research purposes.

4.2.4 Nonlinear Auto-regressive Moving Average Tasks

Nonlinear auto-regressive moving average (NARMA) tasks [202,212-216] are an example of chaotic time series prediction. There are two NARMA tasks commonly used to demonstrate the predictive capabilities of artificial neural systems: NARMA10 and NARMA30. Both tasks take a set of random values $u(k)$ in the interval [0, 0.5] as input with the aim of reproducing a target function $y(k)$. For NARMA10, the target function is given by the recursive formula

$$y_{k+1} = 0.3y_k + 0.05y_k\left[\sum_{i=0}^{9} y_{k-i}\right] + 1.5u_k u_{k-9} + 0.1 \qquad (44)$$

At any point k, the amplitude of $y(k)$ depends on the previous ten random input values u(k) and on the ten previous values of $y(k)$. Hence, the reservoir must have sufficient memory to

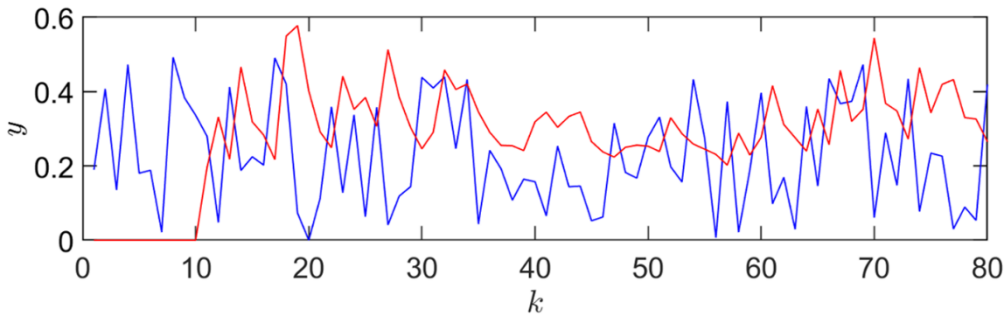


Figure 4.3: Input and target functions for the NARMA10 task. The input $u(k)$ is a sequence of random values in the range [0, 0.5] (blue) from which the target function $y(k)$ is generated by Eq. (44) (red). At every point k, the target function depends on ten previous input and output points.

retain information about ten input and output points in order to construct a good approximation of the target function. The target function for the NARMA30 task depends on the previous 30 input and output values at any point k and is given by the recursive formula

$$y_{k+1} = 0.2y_k + 0.04y_k\left[\sum_{i=0}^{29} y_{k-i}\right] + 1.5u_k u_{k-29} + 0.001 \qquad (45)$$

Given that the NARMA30 task requires information to be retained by the reservoir for 30 points rather than ten points, this is a much more difficult task.

In this thesis, only the NARMA10 task is explored, though the NARMA30 task should indeed be attempted in future work on PASNs. Figure 4.3 gives an example of a sequence of random input values $u(k)$, as well as the corresponding NARMA10 target function $y(k)$ given by Eq. (44). Because y_{k+1} depends on u_{k-9}, the first 10 values of $y(k)$ are zero, Beyond the tenth point, $y(k)$ fluctuates in a chaotic manner. Because $y(k)$ is dependent on a sequence of random positive numbers, there is a possibility that the target function diverges for very long sequences [205]. Hence, sequences of 1000 values are used in this study to ensure that the task provides a significant challenge to the reservoir, but to avoid divergence. The performance of the reservoir at NARMA tasks is quantified by the NRMSE (Eq. (43)) between the reservoir output $\hat{y}_k$ and the target function y_k.

4.2.5 Spoken Digit Recognition

Spoken digit recognition is a practical application where the aim is to correctly classify audio recordings of isolated spoken digits. Examples of this task [204,205] use a subset of the NIST TI-46 corpus [217] which contains ten digits (0…9) each recorded ten times by five different female speakers. The required pre-processing for this task typically involves the use of a cochlear model [218] which converts the audio waveform into frequency channels. Cochlear models vary in their complexity, the simplest being a linear cochlear filter which is approximately equivalent to a Fourier transform. However, the most realistic cochlear filters feature a nonlinear component which have recently been shown to be capable of providing sufficient separation as to successfully perform spoken digit recognition tasks *without* the need for a reservoir at all [219]. Ref. [219] therefore recommends avoiding the use of nonlinear cochlear filters when exploring reservoir performance as they can obscure the true role of the reservoir in the computation.

Following the cochlear filter, time series corresponding to different frequency channels (representing the intensity of each bin in the frequency domain during the recording) are fed into the reservoir where they are nonlinearly transformed. The transformed signals are passed to a readout layer which is a winner-takes-all array of linear classifiers (shown in Figure 4.2), one for each digit. Each classifier is trained to output '1' for its specific digit and '-1' or '0' for all other digits. The reservoir states are passed simultaneously to all of the trained classifiers, and the one which produces the largest positive value is the "winner", resulting in the audio input being classified as the digit corresponding to that linear classifier.

Task performance is measured in two ways: the word error rate (WER) which is simply the relative number of incorrect classifications expressed as a percentage of the total number of testing examples; and the margin, which denotes the smallest difference in the output values of each classifier in the output layer.

4.2.6 Santa Fe Laser

Santa Fe laser is a one-step time series prediction task. The input data originates from an experimental measurement of the intensity of an 81.5 μm $14NH_3$ cw (FIR) laser which exhibits Lorentz chaotic dynamics [220]. The task is to predict the next value in the sequence for every time step. Thus, the target function is equivalent to the input sequence shifted forward by one time step. An example of the Santa Fe laser data set is shown in Figure 4.4 (a), and a zoomed in plot of the input and target sequences in Figure 4.4 (b) [205]. As in the case of the waveform discrimination and NARMA tasks, performance is characterised by the error between the reservoir output and the target function; often expressed as the NRMSE (Eq. (43)).

4.2.7 MNIST Image Classification

The MNIST (Modified National Institute of Standards and Technology) database [221] is a database of images of handwritten digits (0…9) which is commonly used for training image processing systems [222] and in training and testing machine learning algorithms [223]. It is derived from re-mixing the original NIST database [224] testing and training sets, which were from American Census Bureau employees and American high school students respectively. The qualitative differences between the training and testing sets made classifiers

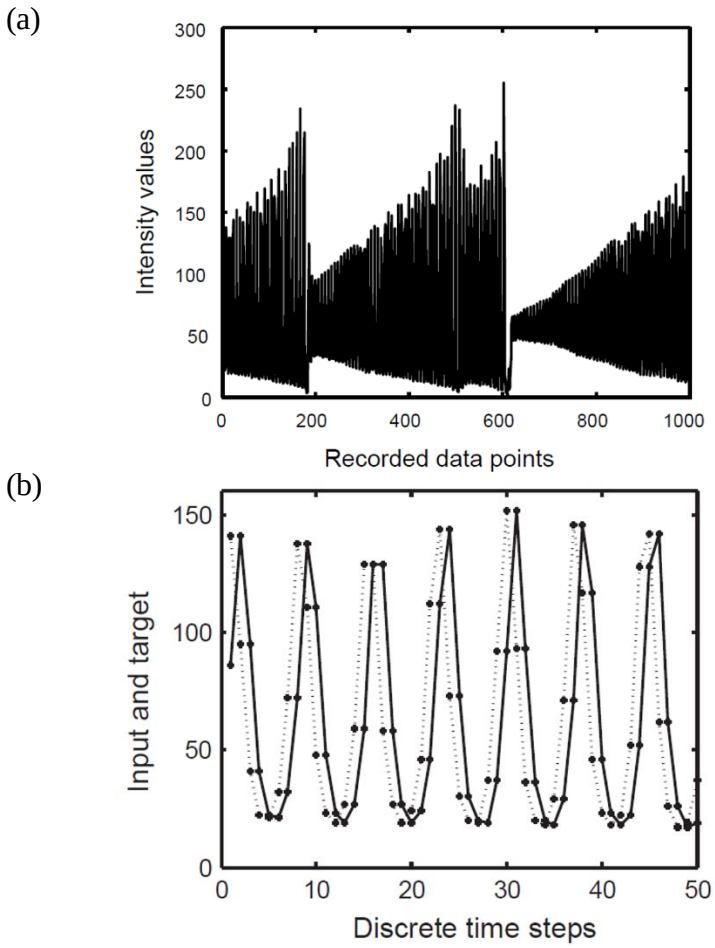


Figure 4.4: Santa Fe laser data. (a) An example of a chaotic time series measurement from the Santa Fe laser. (b) Points connected by dotted line is a subset of the data shown in (a), while the points connected by the solid line represent the corresponding target function; equivalent to the original time series shifted forward by one step. The task is to predict the next value in the chaotic sequence at every time step. [205]

difficult to train on the original NIST database, an issue which MNIST avoids by mixing the two datasets. Additional modifications include normalizing the images to 28×28 pixels and anti-aliasing the original black and white images which introduced greyscale levels. In total, the MNIST database contains 60,000 training images and a further 10,000 testing images.

Hierarchically organised deep convolutional neural networks (see Section 1.2) are the best performing architectures at classifying the MNIST dataset at an error rate of only 0.23% [225], surpassing the performance of support vector machines used by the creators of the MNIST database (0.8% error rate) [221]. In 2017, an extended MNIST database (EMNIST) was published which contains hand-written letters in addition to the original MNIST data [226]. EMNIST contains 240,000 training and 40,000 testing images of alphanumeric characters.

Classification of the MNIST dataset typically involves training a neural network or neuromorphic architecture with an output layer of ten linear classifiers using the winner-take-all approach (Figure 4.2) where each linear classifier is trained to output 1 for one of the ten digits, and -1 (or zero) for all others. For each image, the classifier which outputs the largest positive value is the winner and determines the class to which the image should belong. MNIST has been used to demonstrate classification ability in neuromorphic architectures including those based on reservoir computing [206,227-229]. Although not explored in this thesis, successful classification of the MNIST database would provide strong evidence of the computational power of PASN-based architectures and it is therefore recommended for exploration in future works.

4.3 Time Delay Reservoir Computing Overview

Section 4.1 gave an introduction to TDRC and described the ability of a single time-multiplexed single nonlinear dynamical node (NDN) to emulate a conventional reservoir of spatiotemporally correlated nodes. This section gives a detailed overview of the TDRC framework in terms of its three layers: Section 4.3.1 introduces the input layer and details the required pre-processing of the input signal, Section 4.3.4 describes the hidden layer and the key parameters which determine the connectivity structure of the virtual reservoir, and Section 4.3.5 elucidates the training of the output layer. The explanations given throughout this section are supported by examples of the sine-square waveform discrimination task (Section 4.2.2) using TDRC implemented in software with the Mackey-Glass NDN. The descriptions in this section are adapted from Ref. [205] which provides a comprehensive explanation of the TDRC framework.

4.3.1 Time Scales

There are several important time scales which must be explicitly defined in order to understand the TDRC framework. The first is the time step τ which is equal to the length of the delay line. Each time step is denoted by the index k. It is important to distinguish this definition of a time step from a sampling interval; two terms which are commonly used

interchangeably. During each time step τ, the state of the NDN is sampled N times at regular intervals θ, such that $\tau = N\theta$. Each sample is denoted by the index i such that $i = 1, ..., N$. These N samples make up the N virtual node states which describe the state of the virtual reservoir at some time step k (discussed further in Section 4.3.4). There is one input value $u(k)$ for each time step k which is injected into the NDN for the duration τ. Correspondingly, there is one output value $\hat{y}(k)$ for each time step k which is created by linearly combining all N of the virtual node states sampled during that time step.

Time is, in principle, a continuous variable denoted by $t_{k,i}$. The subscripts indicate that time is measured both in time steps and sampling intervals. For example, the state of the i^{th} virtual node at the k^{th} time step is sampled at $t = k\tau + i\theta$.

The final important time scale is the response time of the NDN which is characterised by a time constant T. The response time of the NDN describes how long it takes the node to reach a steady state after being subject to some new input. The convention adopted in this thesis is to normalize time such that $\theta = 1$ and then describe the other time scales (τ, T) in terms of θ.

4.3.2 Nonlinear Dynamical Nodes

Nonlinear dynamical nodes (NDNs) are objects, which may or may not be physical, that exhibit some nonlinear dynamical response to external stimulation. An unphysical NDN could be an equation or model (e.g. the Mackey-Glass Oscillator (MGO) in Section 5.1) which is implemented numerically on a computer. Conversely, a physical NDN could be a real system which is being physically stimulated and its response measured. This thesis explores both unphysical and physical NDNs. The former are referred to as *numerical NDN models*, while the latter are referred to as *physical NDNs*. Explicitly, the Tunnelling Regime NDN (TRNDN) and Switching Regime NDN (SRNDN) introduced in Sections 5.2 and 5.3 respectively, are numerical models based on real PASN behaviour. The experimental implementation of TDRC in Chapter 6 uses a physical PASN in the tunnelling regime and is thus referred to as a *physical TRNDN*.

4.3.3 Input Processing

In a conventional RC scheme, the reservoir is comprised of many interconnected nodes, as shown in Figure 4.1 (a). The input layer has many input nodes which are connected to the nodes of the reservoir via weighted connections (blue arrows). Input signals are fed simultaneously into the reservoir through these input connections. Conversely, in TDRC the reservoir (called a virtual reservoir) is emulated by time multiplexing a NDN: virtual nodes are points in time along the delay line, rather than points in space like the nodes in a conventional reservoir. It is therefore not possible to inject input signals to all nodes of the virtual reservoir simultaneously, and the input signals must be "flattened" in a pre-processing step. This involves a sample and hold operation and a masking operation which are detailed below.

The input layer takes a discrete input $u(k)$, which in the case of the waveform discrimination task is a balanced random sequence of sine and square waveforms of equal period and amplitude. $u(k)$ is transformed into a piecewise continuous function $I(t)$ by sampling and holding $u(k)$ for a duration τ. This duration τ is the length of the delay line and relates $u(k)$ and $I(t)$ by $I(t) = u(k)$ for $\tau k \leq t < \tau(k+1)$. An example of $u(k)$ and the corresponding $I(t)$ are shown for a single sine wave in Figure 4.5. The reason for this sample and hold operation is so that each node in the virtual reservoir receives the same input signal at each time step, just as each input node would receive the same input signal at each time step in a conventional reservoir. In TDRC each time step k is equal to the length of the delay line τ. The example in Figure 4.5 shows a sine wave sampled eight times per waveform period meaning that the period of the waveform is 8τ.

The N virtual nodes in the delay line are defined by multiplying $I(t)$ by the piecewise continuous binary mask function $M(t)$ which is constant over the period θ (where $\theta = \tau/N$) and periodic over the period τ. Figure 4.6 shows the functions $I(t)$ and $M(t)$, as well as the resulting function $J(t)$, which is the pre-processed input to be injected into the NDN. The mask serves two purposes: firstly, it emulates the role of input connection weights in conventional RC (blue arrows in Figure 4.1 (a)) by multiplying the input to each virtual node by a different value (though only values of ± 1 are used here whereas conventional RC input connections can be float values within this range). Secondly, and perhaps more importantly,

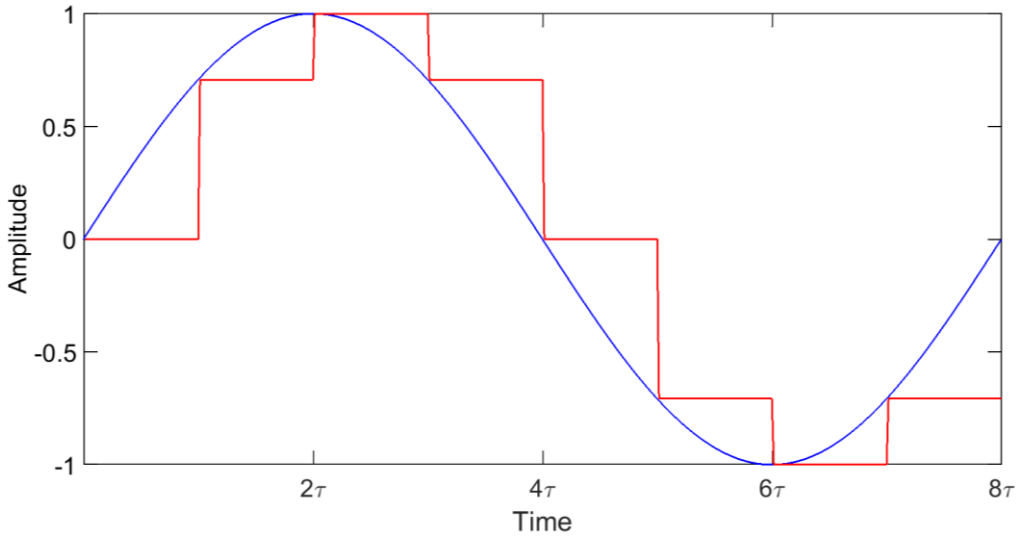


Figure 4.5: Sample and hold operation. The raw input signal $u(k)$ (blue) and the corresponding $I(t)$ (red) resulting from a sample and hold operation.

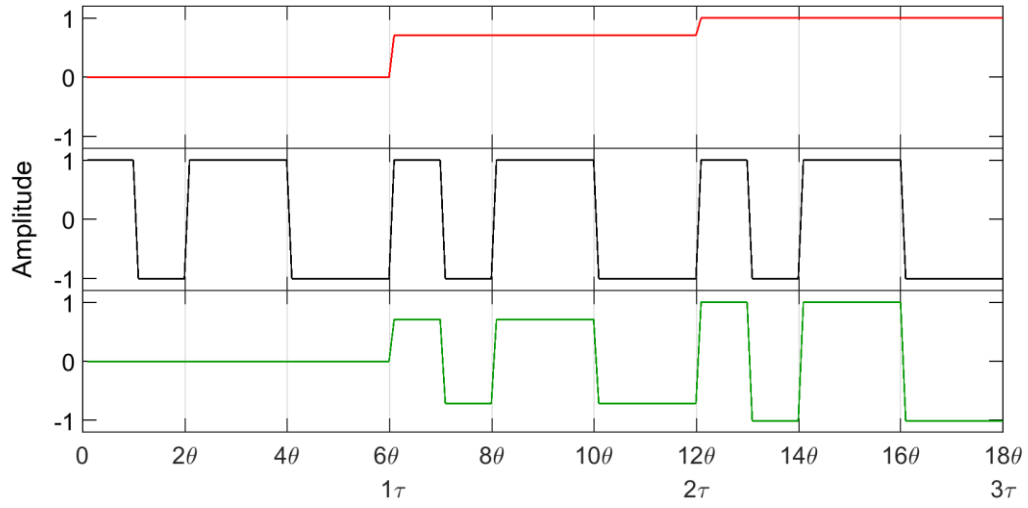


Figure 4.6: Masking procedure. Top: A zoomed view of the same $I(t)$ trace as in Figure 4.5. Middle: A 6-bit binary mask, $M(t)$, which is periodic over τ and constant over θ. Bottom: The pre-processed input signal to be fed to the hidden layer, $J(t) = I(t)M(t)$.

the mask keeps the NDN in a transient regime by choosing the period θ such that $\theta \lesssim T \ll \tau$, where T is the time scale of the nonlinear node [230]. By setting $\theta < T$, the node is never allowed to reach a steady state, and information can pass from one virtual node to the next creating an interconnection structure (in time) which emulates the spatial connectivity between nodes in a conventional reservoir. The temporal connectivity structure of the virtual reservoir is discussed in Section 4.3.4.

4.3.4 Hidden Layer

4.3.4.1 Increasing Dimensionality

The hidden layer is responsible for the higher-dimensional transformation of the input signal. In conventional RC approaches, this transformation is achieved by feeding input signals into the reservoir where each node nonlinearly transforms the signal it receives and passes it to other nodes in the reservoir. At each time step, each node receives (i) signals from the input layer, (ii) signals from other nodes in the reservoir, and (iii) signals from itself from the previous time step via recurrent connections. At each node in the reservoir, these three types of signals are combined and nonlinearly transformed to determine the state of that node (represented by a scalar value) at each time step. The state of the node is the signal which is then transmitted at the next time step to: other nodes in the reservoir via the weighted node-to-node connections; the same node via recurrent connections. The state of a reservoir of N nodes can therefore be described by a $1 \times N$ array of node states at each time step. Because the state of each node depends on the input signal, the previous states of the other nodes, and on its own previous state, the state of the reservoir can be considered a higher dimensional representation of the input signal.

For example, in the task of waveform discrimination, the input signal might consist of a sequence of sine and square waveforms: a vector of K amplitude values where K is the total number of time steps in the sequence. When injected into a reservoir of N nodes, this $1 \times K$ input signal is transformed into an $N \times K$ array of node states, the evolution of each node being described by a $1 \times K$ array of states. The reservoir has therefore increased the dimensionality of the input signal by a factor of N. These N representations are correlated because the state of each node depends on the states of the other node. This correlation is an essential property of the reservoir output, because it allows the N representations of the input

signal to be recombined at the output layer in a useful way (Section 4.3.5). If the node states did not depend on one another, then the N representations of the input signal would be independent transformations, not corresponding to a single higher dimensional space, but rather to many independent spaces of the same dimensionality as the input signal. Higher dimensional transformations increase the separability of different classes of inputs [60], so it is crucial that the reservoir projects the input onto a single high-dimensional space, rather than many independent low-dimensional spaces. The connectivity between the nodes in a reservoir is therefore a vital component of an effective RC implementation.

The *virtual reservoir* in TDRC must produce the same higher-dimensional transformation as described in the above paragraphs. The N virtual nodes are points in time along the delay line, at which the state of a single NDN is sampled. In physical implementations, the state of the NDN is defined as the value of some observable quantity of the physical NDN. This observable is called the *explanatory variable* $x(t)$. Each loop of the delay line corresponds to a single time step τ (see Figure 4.1), during which N samples of $x(t)$ are taken. Each of these samples represents the state of a virtual node. The interval between each sample is θ (such that $\tau = N\theta$) during which time the NDN is responding to the input it is receiving, evolving to the next virtual node state.

The connectivity between virtual nodes must therefore be temporal in nature: the state of the NDN at $t = n\theta$ must depend on the state of the NDN at $t = (n-1)\theta$, meaning that the NDN must possess some memory. If the NDN responds to a new input with some time constant T, then the state of the NDN after receiving the input depends on both the new input, and on its previous state, for a short while after the new input is applied. It is the relationship between the time constant T and the sampling interval θ which determines the connectivity between virtual nodes, as shown in Figure 4.7. If the state of the NDN is sampled at short intervals relative to its time constant ($\theta \ll T$), then the state of the NDN depends mostly on its previous state, having had insufficient time to respond fully to the new input (green in Figure 4.7 (a)). Conversely, if the NDN is sampled at intervals much longer than its time constant ($\theta \gg T$), then the state of the NDN depends only on the new input, its response having completely saturated (red in Figure 4.7 (a)). It is only when the sampling interval is of similar duration to the time constant of the NDN ($\theta \sim T$) that the state of the NDN depends at all times on both its previous state, and on the new input (black line in Figure 4.7 (a)). This

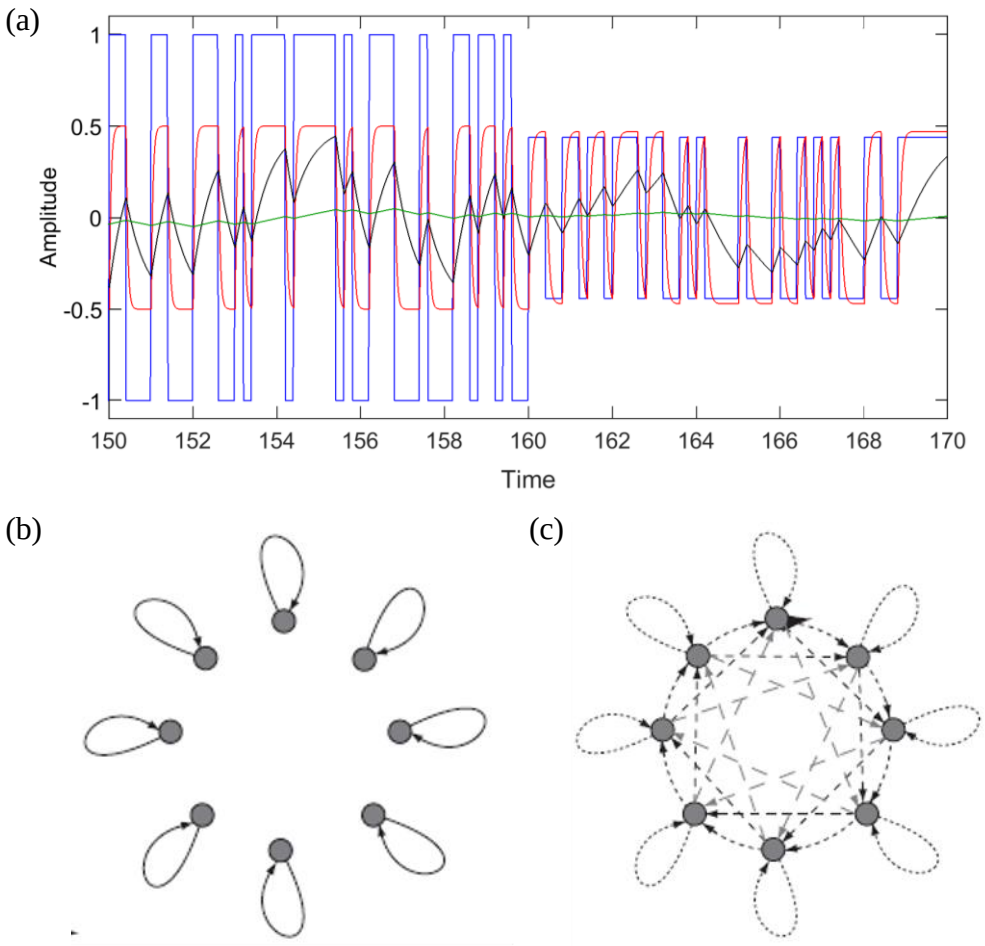


Figure 4.7: θ/T determines the connectivity between virtual nodes. The pre-processed input signal, $J(t)$ (blue), and the response of a Mackey-Glass NDN for different ratios of θ/T: $\theta \gg T$ (red) results in a saturated NDN response leading to small variation in the virtual node states, $\theta \ll T$ (green) produces a highly damped output with approximately linear dynamics, and $\theta \lesssim T$ (black) produces a rich nonlinear transformation of the input signal as the node is kept in a transient regime. (b, c) Illustrations of the connectivity structure between the virtual nodes. (b) In the case where $\theta \gg T$, the saturated NDN response (red in (a)) means that the virtual node states are independent and depend only on their own previous states due to the delayed feedback mechanism. (c) In the case where $\theta \lesssim T$ (black in (a)), the NDN is in the transient regime and the virtual node states are dependent on one another. The different dashed lines represent the connection strength, smaller dashes indicating stronger connections. Nodes which are directly adjacent have stronger connections, and connection strength decreases between nodes which have larger separation along the delay line. (a) was generated using the MGO model (Section 5.1); (b, c) Reproduced from Ref. [202] under CCL.

is called the transient regime, as every virtual node state corresponds to the transient response of the NDN to the present input. In this regime, the state of each virtual node is dependent on the states of the virtual nodes which precede it in the delay line. Hence, keeping the NDN in a transient regime provides temporal connectivity between virtual nodes, and the strength of the connectivity is determined by the relative values of θ and T, as illustrated in Figure 4.7 (b) and (c).

The fact that the NDN must remain in the transient regime elucidates the importance of the role of the masking function $M(t)$ which is applied to the input signal at the pre-processing step. $M(t)$ is constant over θ, during which time the NDN is responding to a constant input and the choice of θ/T determines the extent to which the NDN responds. The exact sequence of binary values in $M(t)$ determines the connectivity structure between the virtual nodes, equivalent to the randomly weighted connections between nodes in a conventional reservoir [230]. So, it is the masking function which is responsible for keeping the NDN in the transient regime and providing connectivity between the virtual nodes.

For the TDRC implementations in this thesis, time is normalised such that $\theta = 1$ and the time constant T is then expressed in terms of θ. The relationship between T and θ is extensively explored as a key parameter for both numerical and experimental TDRC implementations (Chapter 5 and Chapter 6).

4.3.4.2 Nonlinearity

Section 4.3.4.1 described the mechanism by which a virtual reservoir projects an input signal onto a higher dimensional space. This is the core role of the hidden layer and enables successful separation of different classes of inputs. However, this higher dimensional representation is only useful if the transformation of the input signal is *nonlinear*.

The output of RC is generated at the output layer (Section 4.3.5) by creating a linear combination of the N representations of the input signal (one produced by each output node in the reservoir). If these N representations are simple linear transformations of the input signal, then any linear combination of them will result in the equivalent of a single linear transformation of the input. In this case a true higher-dimensional transformation has not been achieved: each of the N representations of the input signal are simply scaled versions of one another and reduce to a single, low-dimensional representation. Consequently, the RC output must follow the same evolution as the input signal, and nonlinearly separable classes (e.g. XOR, Section 4.2.1) cannot be distinguished. If, however, each node in the reservoir

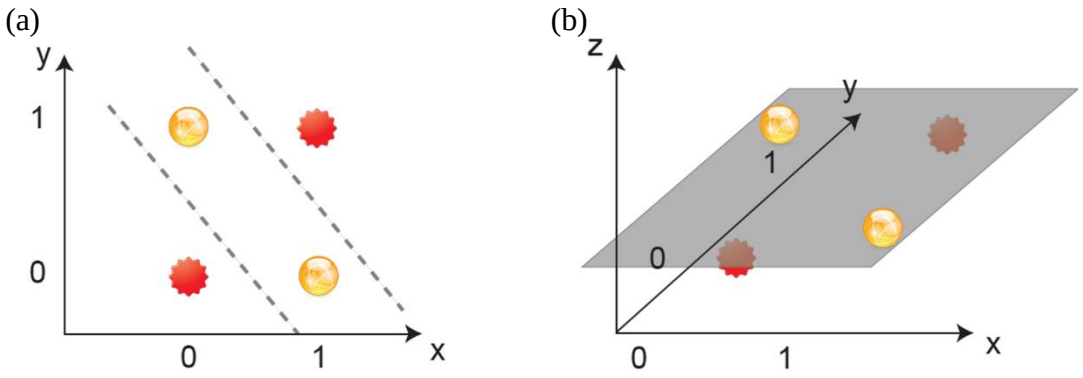


Figure 4.8: Linear separability results from nonlinear higher-dimensional projection. (a) The XOR problem in a two-dimensional space: yellow spheres represent the true response, while red stars represent the false response. The two classes of response cannot be separated by a single straight line. (b) Projection to a higher dimensional space (i.e. adding the *z*-dimension) allows the two classes to be separated by a linear 2-D plane, only if the projection is *nonlinear*. A linear projection onto a 3-D space would mean that each sphere/star had the same *z*-value, and the classes would remain inseparable by a linear 2-D plane. Reproduced from Ref. [202] under CCL.

performs a nonlinear transformation on the signal it receives, the N representations of the input signal can be combined linearly to produce an output which follows a different time evolution i.e. one which cannot be produced by a linear transformation. Only then can nonlinearly separable classes be distinguished.

For example, Figure 4.8 illustrates how linear separability is achieved using a nonlinear projection onto higher-dimensional space. The two response classes of the XOR problem (yellow spheres: true; red stars: false) cannot be separated by a single straight line in 2-D. In 3-D however, the two classes may be separated by a single linear plane, but only if the transformation is nonlinear i.e., they have different positions along the newly introduced *z*-dimension. If the transformation was linear, then each sphere/star would have the same z-value, and the classes would remain inseparable.

In conventional RC, the nonlinear transformation is produced by a nonlinear activation function at each node in the reservoir: All of the inputs to a node are summed and passed through the activation function to produce the output of that node which is then distributed to other nodes in the reservoir at the next time step. In TDRC, the NDN must exhibit a nonlinear response to the input signal to generate the required nonlinear transformation.

4.3.4.3 Recurrence

Sections 4.3.4.1 and 4.3.4.2 explain how the connectivity between nodes in a reservoir, and the nonlinear transformations taking place at each node, produce a higher dimensional

representation of the input signal which can be used to perform classification tasks. Another important feature of a reservoir is recurrent connectivity: connections which span different time steps.

In conventional RC, the nodes feature explicit connections which pass the state of each node to the same node at the next time step. This gives the reservoir memory and allows for the classification of *sequences* of inputs (Section 1.2.2). In TDRC, where the nodes are points in time along the delay line, the memory of the NDN, which is characterised by the response time T, facilitates the temporal connectivity between the nodes in the virtual reservoir. As this connectivity is an analogue of the spatial connectivity between nodes in conventional RC, the recurrent connections in a virtual reservoir must be supplied by another mechanism. A delayed-feedback loop is used for this purpose. By combining the previous state of the virtual reservoir at each time step with the new input signal, the state of each virtual node is dependent on its own previous state, fulfilling the role of the recurrent connections in RC. The role of delayed feedback and its effect on reservoir performance is explored in both numerical and experimental implementations of TDRC (Chapter 5 and Chapter 6).

4.3.5 Output Layer and Training

The output layer and training procedures used in TDRC are identical to those of conventional RC. When a $K \times 1$ sequence of inputs $J(t)$ is injected into the reservoir, a $K \times N$ matrix of node states X is produced. Each row is the state vector of the reservoir at one time step k where $k \in [1, K]$, and each column represents the response of one node as a function of time. This matrix is called the *predictor matrix* or *observation matrix* as it contains observations of the virtual reservoir (values of the explanatory variable which define virtual node states) which are used to predict the target function. The $K \times N$ predictor matrix has a constant bias term[13] added to it so that it becomes $K \times (N + 1)$.

The output layer is a map from the state of the reservoir $X(k)$ to the desired output i.e. the target function $y(k)$. This map is achieved by training a set of weights w_i where $i \in [1, N + 1]$ so that a linear combination of the $N + 1$ columns of X, $\hat{y}(k)$, can be constructed to approximate the target function $y(k)$:

[13] The bias term allows for the vertical offset of $\hat{y}$ to be fitted to that of the target function y by adapting the corresponding weight w_{N+1} during training.

$$\hat{y}(k) = \sum_{i=1}^{N+1} w_i x_i \approx y(k) \tag{46}$$

The weights w_i are chosen to minimize the mean square error between $\hat{y}(k)$ and $y(k)$ i.e. $\|Xw - y\|^2$. This is achieved by linear regression, using

$$w = (X^{\dagger} y)^T \tag{47}$$

where † denotes the Moore-Penrose pseudoinverse which is used to find a solution to a system of linear equations which may not have a unique solution. This training process is carried out using some subsection of the dataset called 'training data' (usually 50% of the dataset) and results in a set of weights being chosen for the output layer. A subsequent testing process is then performed on the remainder of the dataset (which was not used in the training process) using the predetermined weights. If the reservoir has been successfully trained, the output $\hat{y}(k)$ should approximate the target function $y(k)$ for the testing data with similar accuracy to that observed in the training process.

In the case of *overfitting*, where the training successfully produces weights that can generate a good approximation of $y(k)$ for the training data, but fails to do so for the testing data, regularization may be used requiring an additional *validation* process. In this thesis, the method of Tikhonov regularization, otherwise known as ridge regression, is used to circumvent overfitting [51,52]. Unlike simple linear regression, which seeks to minimize the mean square error $\|Xw - y\|^2$, ridge regression aims to minimize the cost function

$$\|Xw - y\|^2 + \|\lambda w\|^2 \tag{48}$$

where λ is a constant called the ridge parameter which has the effect of penalizing large weights. The larger the value of λ, the greater the penalty, and the smaller the magnitude of the chosen weights. When using ridge regression, the training process involves choosing a wide range of λ values and obtaining a set of weights w which minimizes Eq. (48) for each λ value. This necessitates the validation procedure, whereby a set of validation data which is exclusive to the training and testing data is used to choose the best λ value, and therefore the best set of weights. This is achieved by selecting the value of λ which produces the lowest normalized root mean square error (NRMSE) between the constructed linear combination $\hat{y}(k)$ and the target function $y(k)$ for the validation dataset.

The TDRC procedure used in this thesis is summarised in Figure 4.9, using the sine/square waveform discrimination task (Section 4.2.2) as an example. Figure 4.9 (a) shows

the discretised sequence of sine and square waveforms, each cycle of which has eight time steps denoted by the red dots. The top panel of Figure 4.9 (b) shows a subsection of $J(t)$ containing a sine and square waveform after the mask has been applied. The mask has $N = 12$ bits, so there are 12 virtual nodes along the delay line. The bottom panel of Figure 4.9 (b) shows the response of a Mackey-Glass NDN (Section 5.1) to the input stream in the top panel. Figure 4.9 (c) shows the NDN response over one time step τ, which is equivalent to one loop around the delay line, during which the state of the NDN is sampled N times at intervals of θ. These N samples represent the state of the virtual reservoir during that time step which is mapped to the target function (Figure 4.9 (d)) by combining the virtual node states into a weighted linear sum. One set of N weights are found by minimising the difference between the reservoir output $\hat{y}(k)$ (red circles) and the target function $y(k)$ orange line for the training data. That set of weights is then applied to every time step in the testing data, in the manner shown in Figure 4.9 (c) and (d). If the weights have been correctly determined, the virtual reservoir is trained and the reservoir output $\hat{y}(k)$ should be a good approximation of the target function $y(k)$.

For sequence classification tasks, such as waveform discrimination, one final step called "output squashing" is performed. Each time step k produces one point in $\hat{y}(k)$ and each waveform contains 8 time steps[14] (see Figure 4.9 (a)). However, each waveform belongs to only one class: sine or square. Thus, it is necessary to "squash" the eight values into a single value which can be used to assign the predicted class for that waveform. This is performed by taking the mean of $\hat{y}(k)$ over each waveform period i.e. 8 time steps, which is denoted as $\langle\hat{y}\rangle$. In this thesis, if $\langle\hat{y}\rangle$ is positive then the waveform is classified as a sine wave, and if $\langle\hat{y}\rangle$ is negative, the waveform is classified as a square wave. This is similar to using a threshold function (with the threshold at zero) which is a common method of squashing the output [231].

[14] In theory, any number of timesteps could be used for each waveform. In this thesis, the convention is to use 8 time steps per waveform, consistent with Refs. [146,201,209,210].

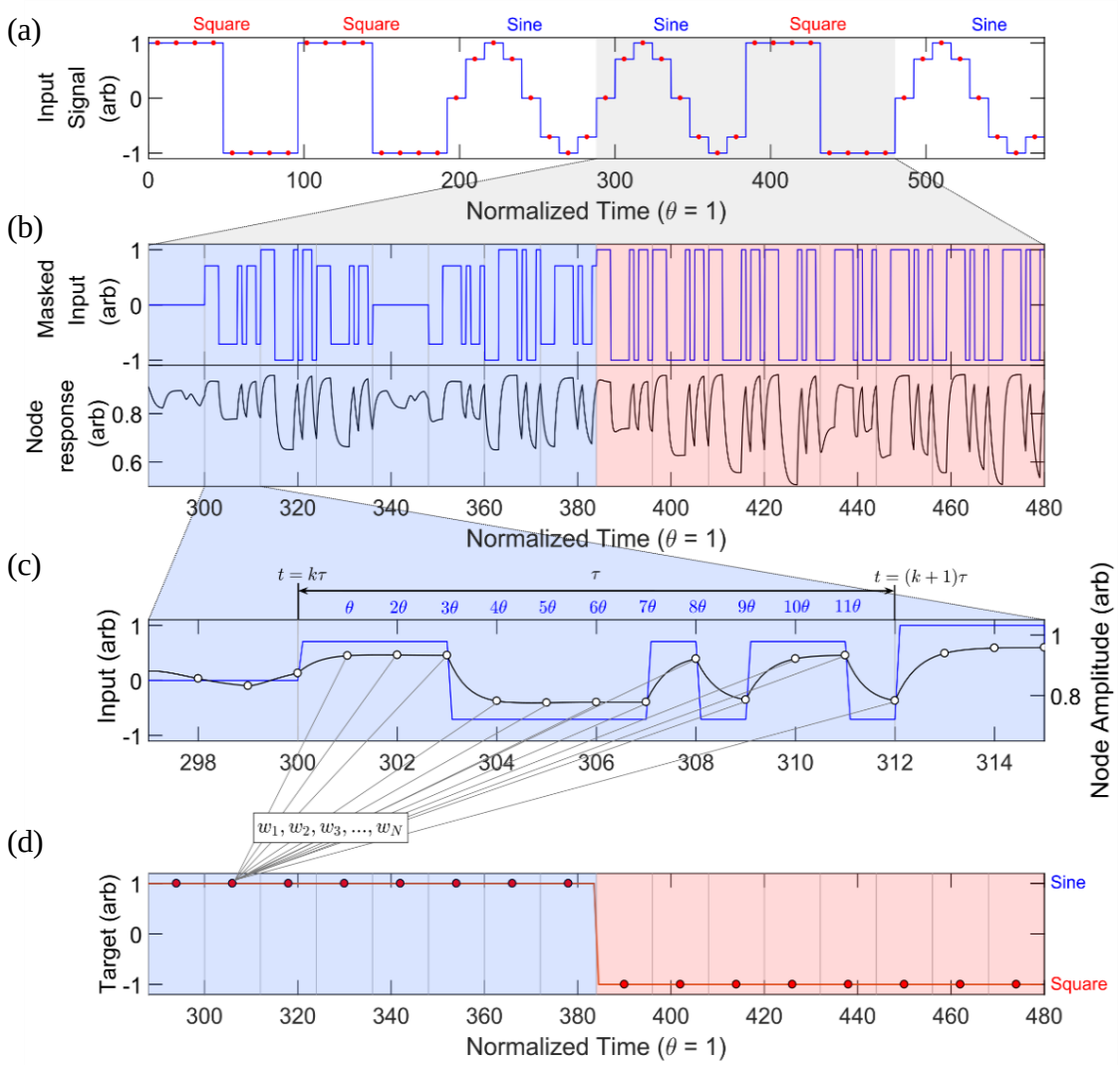


Figure 4.9: Summary of TDRC procedure. An example of sine/square waveform discrimination demonstrating how the target function is approximated by the virtual reservoir output. (a) The discretised input sequence of sine and square waveforms using 8 time steps (τ) per cycle. (b) Top: A subsection of the input sequence $J(t)$ showing the masked waveforms. The mask length is $\tau = N\theta$ where $N = 12$. Bottom: The response of the single NDN to the input sequence. In this case, a Mackey-Glass NDN was used (Section 5.1) with $T = 1\theta$, $\eta = 0.5, \beta = 0.05$ and $p = 1$. (c) A zoomed region of the NDN response showing that the state of the NDN is sampled at intervals of θ, making up the 12 virtual node states in each time step. (d) Each of the N virtual node states are multiplied by a weight w_i, where $i = 1 \ldots N$, such that their linear combination produces the correct value corresponding to the target function shown in red. As this section of data corresponds to the second time step in a sine wave, the target value is one, while for square waves, the target values are minus one. Figure inspired by those in Ref. [146].

The NARMA10 task (Section 4.2.4), which is also explored in this thesis, uses the same procedure as the waveform discrimination task in Figure 4.9. The only differences are that the inputs are random input values $u(k)$ in the interval [0, 0.5] rather than waveform amplitude values, and the target function, which for NARMA10 is given by Eq. (44). Each input value is masked and applied for one time step. As NARMA10 is a time series prediction task rather than a sequence classification task, no squashing of the output is required.

4.4 Related Literature

This section reviews literature regarding the use of the TDRC framework in hardware applications of reservoir computing, a field which has gained increasing interest in recent years. Further details on such implementations, and on those utilizing conventional RC for neuromorphic applications, can be found in the review in Ref. [232] and citations therein.

TDRC, as outlined in Sections 4.1 and 4.3, was first conceptualized in Ref. [205]. Conventional RC requires a network of many nodes in order to generate a high-dimensional representation of input signals. Time-delayed dynamical systems on the other hand can generate high-dimensional patterns [233] requiring only a single NDN which is time-multiplexed to emulate a network of virtual nodes [202]. In neuromorphic implementations of RC, where the reservoir must be facilitated by a physical system, device complexity and power consumption can be largely reduced by the need for only one physical node. Hence, TDRC is becoming a popular choice for novel hardware implementations of RC [51,201,202,204].

4.4.1 Electronic Circuit Implementation

The first hardware implementation of TDRC was an analogue electronic circuit which emulated the Mackey-Glass oscillator (Section 5.1) and featured a delayed feedback mechanism [202]. Two benchmark tasks were used to demonstrate classification and prediction capabilities: spoken digit recognition (Section 4.2.5), and NARMA10 (Section 4.2.4).

The WER of the spoken digit recognition task was reported to be 0.2%, a result which is comparable to the best performing software implementations of conventional RC at the time [234]. A simulation of the analogue circuit was also found to perform well at the spoken digit

recognition task, producing a WER as low as 0.14%. It is typical that simulated systems perform slightly better than their experimental counterparts due to the absence of real-world interference such as electronic noise. In this thesis, a software implementation of TDRC based on that in Ref. [202] is used extensively to explore the parameter space of the TDRC framework.

The NARMA10 task was performed using a simulation of the analogue circuit for different virtual reservoir sizes. The NRMSE values between the reservoir output and the target function for the testing data range between 0.53 (for $N = 20$) and 0.12 (for $N = 400$). The authors note that a NRMSE of 0.4 corresponds to the performance of a linear shift register, and only performance which results in NRMSE < 0.4 is of any real interest. The NARMA10 task was not explored experimentally in Ref. [205], but in Ref. [202] a NRMSE value of 0.15 is reported for the NARMA10 task using the same system.

4.4.2 Photonic Implementations

An experimental opto-electronic implementation of TDRC was realized in Ref. [235] which used an Ikeda nonlinearity [236] and a long spool of optical fibre as the delay line. Optical implementations have attracted significant attention because, like the electronic circuit implementation in Ref. [202], components are commercially available, but the high bandwidth of optical circuitry has positive implications for the processing speed which can be attained with such systems [51].

Several benchmark tasks were used to evaluate the performance of this TDRC implementation, including spoken digit recognition (Section 4.2.5) and Santa Fe laser data prediction (Section 4.2.6). For the spoken digit recognition task, Lyon's cochlear ear filter [218] was used in the pre-processing step and a virtual reservoir of 400 nodes was emulated using a θ/T ratio of 0.217. The WER was found to be as low as 0.04 ± 0.017 %. Recent work by the same group has demonstrated that this impressive classification accuracy can be achieved at rates of ~ 10^6 words per second [207].

Performance at the Santa Fe laser prediction task was characterised by the normalised mean-square error (NMSE), which was reported to be $0.124 \pm (4 \times 10^{-4})$ [235]. It was later demonstrated that the latter result could be significantly improved (NMSE ≈ 0.06) by utilizing a multi-valued mask, rather than the conventional binary mask. This was further improved (NMSE ≈ 0.02) by reducing the quantization noise in measuring the physical

reservoir by oversampling the NDN state and averaging the oversampled signal [237]. The result for this task is comparable to that observed from the noise-free numerical RC implementation in Ref. [238].

Another opto-electronic delay-based reservoir was developed independently in Ref. [201]. This implementation also uses a spool of optical fibre as the delay line but uses the sine nonlinearity of an integrated Mach-Zehnder intensity modulator as the NDN. Performance is assessed by the performance of a virtual reservoir of at four benchmark tasks: sine/square waveform discrimination, NARMA10, nonlinear channel equalization, and isolated spoken digit recognition. For the waveform discrimination task, a NMSE of 1.5×10^{-3} was achieved (100% classification rate). The NARMA10 performance is characterised by a NMSE of 0.168 ± 0.015, which the authors note is similar to the result (NMSE $= 0.15 \pm 0.01$) obtained by a numerical reservoir (also $N = 50$) in Ref. [239].

The same group further demonstrated the computational abilities of their photonic TDRC implementation in Ref. [240] by performing the NARMA10 task, nonlinear channel equalization, and spoken digit recognition. In this work, a quadratic readout layer was used, where the reservoir output given by Eq. (46) is replaced by

$$\hat{y}(k) = \sum_{i=1}^{N} w_i \, |x_i|^2 \approx y(k) \tag{49}$$

For a virtual reservoir of 50 nodes, the NARMA10 task NMSE was reported as 0.062 ± 0.008 for the simulated system and 0.107 ± 0.012 for the experimental implementation. Increasing the number of virtual nodes improved these results to 0.0463 ± 0.0142 and 0.0484 ± 0.0095 respectively. The authors note that their result surpasses that of the best-performing experimental photonic RC implementation at the time [201]. In the nonlinear channel equalization task, an error rate of 0% was obtained for both the simulated and experimental systems, meaning that all of the 60, 000 symbols in the testing sequence were correctly reconstructed. Excellent performance was also demonstrated in the spoken digit recognition task which produced WERs of 0% for both simulation and experiment. The photonic reservoir was further tested by adding random background noise to the spoken digit recordings (3dB SNR) and repeating the task using 500 virtual nodes. Even with the additional noise the reservoir performed well, yielding WERs of $0.6(\pm 0.9)\%$ and $0.8(\pm 0.8)\%$ for the simulation and experiment respectively.

4.4.3 Spin-torque Oscillator Implementations

Recently the TDRC has been implemented using a nanoscale spintronic oscillator (magnetic tunnel junction) as the NDN [204]. When currents flow through a magnetic tunnel junction, they become spin-polarized and generate torques on the magnetizations. This causes microwave frequency oscillations in the magnetic field, which can be measured in the voltage across the junction due to the magneto-resistance effect. The amplitude of these microwave oscillations is a nonlinear function of the current flowing through the junction and changes to the current cause a corresponding change to the oscillation amplitude with a characteristic time constant T. The oscillation amplitude is therefore used as the explanatory variable as it features both the nonlinearity and memory required for TDRC. Interestingly, this implementation features no delayed feedback mechanism, so the virtual reservoir emulated by the magnetic tunnel junction is essentially a feed-forward network (Section 1.2.1).

[204] was the first realization of nanoscale oscillators being used for neuromorphic computing and was shown to achieve state-of-the-art performance at spoken digit recognition (using 400 virtual nodes) and waveform discrimination (using 24 virtual nodes). The advantages of this approach include high stability relative to other attempts at using nanoscale oscillators for neuromorphic computing, and the potential for upscaling by combining many spintronic oscillators into a nonlinearly interacting network. Compared with the photonic implementations, spin-torque oscillators have the advantage of being CMOS compatible nanoscale elements (photonic delay lines are typically large spools of fibre optics cable), but their bandwidth is limited by the oscillator amplitude response time and so the photonic reservoirs are capable of much faster processing.

The WER of the spoken digit recognition task was found to be 0.4% and the waveform discrimination produced a 100% classification rate with a standard deviation in the RMSE of only 1% (though the RMSE values are not given in Ref. [204]). The spoken digit recognition task was performed using two types of pre-processing filters: a linear spectrogram filter and a nonlinear cochlear ear filter [218]. The WER of 0.4% quoted above was found using the latter, while the best WER using the linear filter were around 20%. This highlights the important role of the nonlinear pre-processing step, a fact which is elucidated in Ref. [219] where it is shown that high classification rates can be achieved using the nonlinear frequency filter in complete absence of the spin-torque oscillator. It is hence recommended that linear

frequency filters be used for this task as not to obscure the computational contribution from the neuromorphic architecture under study.

The waveform discrimination task is examined more closely in Ref. [146] where the performance is presented as the RMSE between the reservoir output $y(k)$ and the target function $\hat{y}(k)$. The performance was explored as a function of the magnetic field and DC offset current used to drive the oscillator. They find the NMSE for the optimised parameters to be 0.11 (NRMSE = 0.33).

The same task was further explored in Ref. [210] where the amplitude, frequency and phase of the microwave oscillations are each used as the explanatory variable. The best performance is found for the phase of the oscillations, giving a classification rate of 99.75% (one misclassified waveform in 400), which is more nonlinear than the oscillation frequency (99.5% successful classifications) and less noisy than the oscillation amplitude (99.0% success). It is important to note that these tests were performed without a delayed feedback loop and with a very large θ/T ratio, meaning that the virtual reservoir has no memory of past states and connectivity between the virtual nodes is weak. Consequently, the reservoir was capable of classifying sine and square waveforms only if their amplitudes differed significantly (50% difference). In the non-trivial case, where the waveform amplitudes are the same, it is the *sequence* of input values (which differ for sine and square waves of equal amplitude) which must be used to classify the waveforms. Much lower classification rates were found for equal amplitude waveform discrimination, the best being 82%, though it is not stated which of the explanatory variables is used to obtain this result [210].

In Ref. [241] the same group experiment with the inclusion of a delayed feedback mechanism. The performance at the waveform discrimination task is explored as a function of the operating parameters, and principle component analysis is used to show that the inputs are better separated when the delayed feedback mechanism is included: the best classification rate without feedback was found to be 89.2 %, which was improved to 98.4% with the addition of feedback. This work highlights the importance of recurrence in network architectures which are used to classify temporal inputs.

4.4.4 Summary of Literature Results

The tables in this section summarise the results of several benchmark tasks reported in the literature on different physical TDRC systems. Where possible, several results are also

included from state-of-the-art numerical RC systems for comparison e.g. echo-state networks. Shaded cells denote results from simulations of physical TDRC implementations.

Waveform Discrimination (% correct)		
Reference (case)	N	
	24	50
[51]	97.5% ($N = 25$)	
[201]		100% (0.0387)
[146]	100% (0.33)	
[210] (Unequal amplitudes)	99 %, 99.5%, 99.75 %	
[210] (Equal amplitudes)	82%	
[241]	89.2% → 98.4% by adding feedback	

Table 1: Literature results for the waveform discrimination task. Results are given as % of correctly classified waveforms with NRMSE values in brackets where given. The three values given for Markovic (Unequal amplitudes) correspond to the use of different explanatory variables (Section 4.4.3).

Spoken Digit recognition (WER %)								
Reference (case)	N							
	50	100	150	200	308	388	400	500
[202]							0.2	
[205] (Electronic)	3	1.6	1.2	0.8	0.3		0.14	
[235]							0.04	
[205] (Photonic)							0.02	
[201]				0.4				
[242]			0.6					
[243]						0.014		
[244]							0.05	
[207]							0.04	
[240] (Exp)				0				0.6*
[240] (Sim)				0				0.8*
[204]							0.4	
[238]	7.32	2.96	1.82	1.38				
[234]					0.2			

Table 2: Literature results for the spoken digit recognition task expressed as the word error rate %. The * symbols denote that babble noise (SNR 3dB) was added to the recordings prior to classification to challenge the system.

Santa Fe Laser (NMSE)						
Reference (case)	*N*					
	50	100	150	200	388	400
[205] (Electronic)	0.0228	0.0214	0.0212	0.021		0.019
[235]						0.124
[205] (Photonic)						0.124
[237]						0.02
[243]					0.106	
[230]						0.14
[238]	0.0184	0.0125	0.00945	0.00819		-

Table 3: Literature results for the Santa Fe laser one-step prediction task expressed as the NMSE.

Nonlinear channel equalisation 28dB (error rate)				
Reference (case)	*N*			
	50	100	150	200
[201]	1.30E-04			
[240] (Exp)	0			
[240] (Sim)	0			
[238]	0.0038	0.0021	0.0015	0.0013
[192]	1e-4 - 1e-5			

Table 4: Literature results for the nonlinear channel equalisation task where the SNR is 28 dB, expressed as the error rate (fraction of incorrectly reconstructed symbols).

NARMA10 (NRMSE)								
Reference (case)	*N*							
	20	50	100	150	200	300	400	520
[202]							0.15	
[205] (Electronic)	0.53	0.35	0.27	0.21	0.18		0.12	
[205] (Photonic)							0.22	
[201]		0.41						
[230]								0.134
[240] (Exp)		0.327				0.22		
[240] (Sim)		0.249				0.215	0.103	
[239]		0.39						
[238]	-	0.41	0.31	0.23	0.21		-	
[245]	0.56	0.29	-	-	-		0.099	
[192]								
[234]			0.18					

Table 5: Literature results for the NARMA10 task expressed as the NRMSE.

Chapter 5

Numerical Implementations of TDRC

Before attempting hardware based TDRC using a PASN as the NDN, TDRC was first implemented numerically using several numerical NDN models, in order to gauge the suitability of TDRC as a framework in which to utilize the dynamics of PASN electrical signals. Additionally, there are many options when choosing the explanatory variable $x(t)$ of a physical NDN [210] and so another aim was to identify a suitable explanatory variable.

Section 5.1 presents a numerical implementation of TDRC that uses the Mackey-Glass oscillator [246] as the NDN. This model has been successfully used to perform TDRC [202] and is thus explored to elucidate the TDRC procedure and to provide a basis for comparison with other numerical NDN models in this thesis.

Section 5.2 introduces a NDN model called the Tunnelling Regime NDN (TRNDN) which was designed to mimic the nonlinear conductance-voltage relationship of PASNs in the tunnelling regime i.e. when subthreshold voltages are applied to the network (Section 1.4.4). In this model, input signals are applied as voltages and the explanatory variable is the conductance.

Section 5.3 presents the Switching Regime NDN (SRNDN) which models the bursty switching rate and exhibits critical avalanche dynamics (Section 3.2.3) as observed in PASNs in the switching regime. Ref. [15] shows that the PASN switching rate is a nonlinear function of applied voltage, and so the switching rate is used as the explanatory variable.

The various models introduced in this section are assessed using two benchmark tasks: sine/square waveform discrimination (Section 4.2.2) and NARMA10 (Section 4.2.4). The performance of the different models and parameter choices are compared and used to inform the experimental TDRC approach detailed in Chapter 6.

5.1 Mackey-Glass Oscillator

TDRC is a complex paradigm with many subtleties so the important features of a dynamical node and the important parameter relationships are not readily apparent in the literature, beyond the specific cases in Ref. [205]. For example, Ref. [205] demonstrates that a NDN based on the Mackey-Glass oscillator (MGO) is capable of performing well at a speech recognition task using certain parameters, but how would that model perform using different parameters, or at a different classification task? What if the nonlinearity is different? Is the recurrence provided by the delayed feedback mechanism essential to good performance on all tasks? These types of questions are not typically elucidated in the literature on TDRC and make implementing TDRC with novel systems initially challenging. Hence, the Mackey-Glass model is implemented as a numerical NDN in order to explore the parameter space of the TDRC framework (N, T, θ etc), and to use as a starting point for exploring different NDN models which are more closely related to the PASN devices which are the focus of this thesis.

MGO Model Description

The MGO [246] is a delayed feedback differential equation which was initially proposed to model oscillations in blood-cell populations. Because of its ability to generate a range of dynamic outputs including oscillations and chaos, it has become a standard method of testing algorithms for the quantitative characterization of chaotic dynamics [235]. Additionally, the electronic TDRC implementation in Ref. [202] was modelled on the MGO due to its simp-

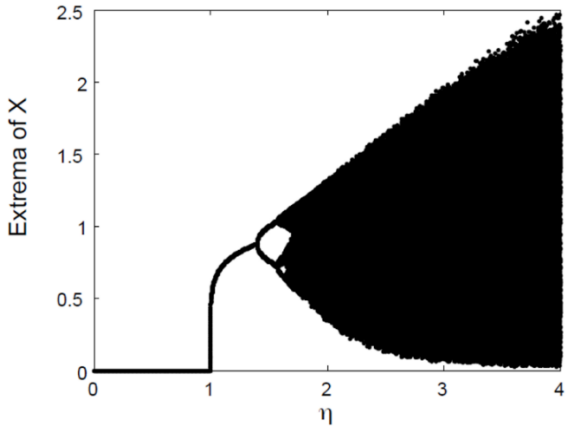


Figure 5.1: Orbit diagram for the MGO. Depending on the feedback strength η, the MGO operates in a range of dynamical regimes from a zero fixed point to a non-zero fixed point, limit cycles, and deterministic chaos. ($\tau = 80, \gamma = 0, p = 6.88$) [205]

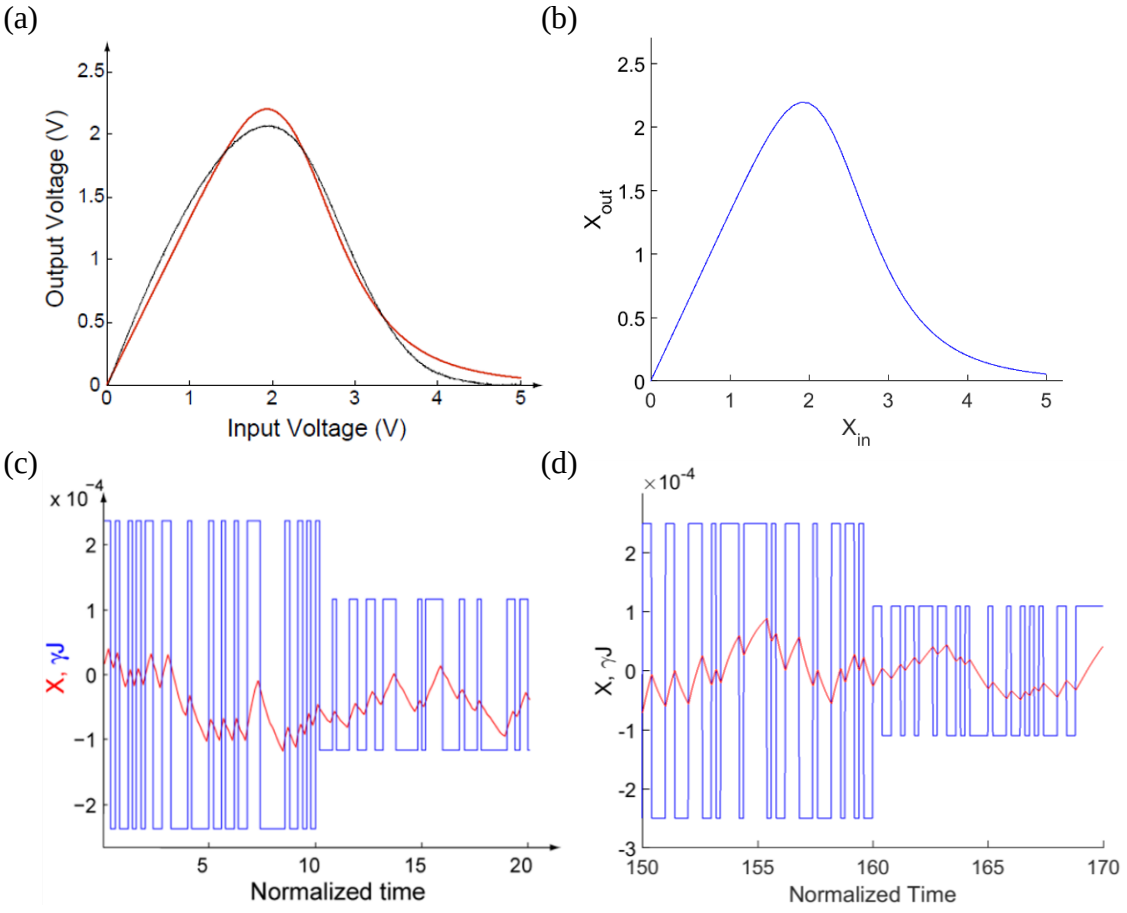


Figure 5.2: Verification of the Mackey-Glass nonlinearity. The Mackey-Glass nonlinearity shape for (a) the analogue circuit (black) and corresponding fit (red) from Ref. [205], and, (b) our implementation (blue). The red and blue curves are identical. (c) The MGO response X to a masked input sequence J from Ref. [205], and, (d) from our implementation. The responses are qualitatively similar with quantitative differences being due to the different input sequences and the unknown initialization of the response in (c).

-licity, noise robustness, and tunability.

Formally, the MGO is represented by the equation

$$\frac{dx}{dt} = \beta \frac{x_\tau}{1 + {x_\tau}^n} - \gamma x \tag{50}$$

where β, γ, n are real numbers and x_τ represents the explanatory variable x at a time $(t - \tau)$. This representation of the MGO depends only on x, not on any external input. Therefore, in accordance with [205], the specific form of Eq. (50) used in this thesis is

$$\frac{dx}{dt} = \frac{1}{T} \frac{\eta(x(t-\tau) + \gamma J(t))}{1 + (x(t-\tau) + \gamma J(t))^p} - x(t) \tag{51}$$

where x is the explanatory variable, T is the time constant of the oscillator, τ is the delay period, and η, γ and p are real constants which in Ref. [202] were fixed at $0.5, 0.05$ and 1 respectively. γ determines the strength of the input signal $J(t)$ relative to the delayed explanatory variable $x(t - \tau)$, while η controls the delayed feedback contribution. The effect of varying η is demonstrated via the orbit diagram in Figure 5.1 [202,205] for $\gamma = 0$. The dynamics range from fixed points, to limit cycles, to deterministic chaos. In Ref. [205], the value of $\eta = 0.5$ was chosen so that the MGO would operate at a stable fixed point in the absence of external input ($\gamma = 0$), but with external input ($\gamma > 0$) will exhibit complex dynamics. The same parameters of the MGO used in Ref. [202] are adopted in this thesis.

The implementation of the MGO is verified by comparing both the nonlinearity (Figure 5.2 (b)) and the MGO response to external input (Figure 5.2 (d)) with those of [205] (Figure 5.2 (a) and (c) respectively). The nonlinearities (red in Figure 5.2 (a) and blue in Figure 5.2 (b)) are identical, while the response X is qualitatively similar. The quantitative differences in X arise only due to the different input sequences (blue) and the different initial states. Hence this implementation of the MGO is consistent with [205].

To investigate the role of the delayed feedback mechanism, an alternative version of the MGO *without delayed feedback* is also explored. In this case, the MGO is characterised by the equation

$$\frac{dx}{dt} = \frac{1}{T} \frac{\eta(J(t))}{1 + J(t)^p} - x(t) \tag{52}$$

where η and p retain their values of 0.5 and 1 respectively.

5.1.1 Waveform Discrimination using the MGO

This section presents results of the waveform discrimination task (Section 4.2.2) using the MGO as the NDN in a TDRC framework. The waveform discrimination task involves classifying complete sine and square waveforms of equal amplitude within a random sequence of waveforms. The trained virtual reservoir should output a binary value indicating whether the incident waveform was a sine or a square. The convention used in this thesis is '1' for a sine wave and '-1' for a square wave. The task was performed both with and without delayed feedback, and for a wide range of different N and T.

Example Result with Delayed Feedback

Figure 5.3 presents an example result of the waveform discrimination task using the MGO *with* delayed feedback for $N = 100$ and $T = 5\theta$. Figure 5.3 (a) contains a subsection of the input sequence $J(t)$ which shows six time steps (i.e. 6 τ) during a sine wave. Figure 5.3 (b) shows the response of the MGO (i.e. the explanatory variable $x(t)$) corresponding to the input shown in Figure 5.3 (a). Figure 5.3 (c) shows the final output of the virtual reservoir $\hat{y}$ (blue) which is the linear combination of weighted virtual node responses given by Eq. (46). The result in Figure 5.3 (c) shows the result from 400 waveform cycles: 200 waveform cycles were used to train the weights w_i and a further 200 waveform cycles were used to test the trained system. In Figure 5.3 (c), $\hat{y}$ is compared with the target function y (red), but because $\hat{y}$ is so similar to y, the red line cannot be seen on this scale. Hence, a zoomed plot of Figure 5.3 (c) is shown in Figure 5.3 (d) showing only 50 waveforms. Here $\hat{y}$ can be seen fluctuating around y with a very small error. This error is expressed as a NRMSE of 1.41×10^{-6} which in this case is the same for the training and testing data.

The black line in Figure 5.3 (d) represents the squashed output $\langle\hat{y}\rangle$, which is the average of $\hat{y}$ over each waveform period. The predicted class of each waveform is decided by the sign of $\langle\hat{y}\rangle$: when $\langle\hat{y}\rangle$ is positive, the corresponding waveform is classified as a sine wave, and when $\langle\hat{y}\rangle$ is negative, the corresponding waveform is classified as a square wave. The results in Figure 5.3 show that the MGO correctly classified 100% of the waveforms.

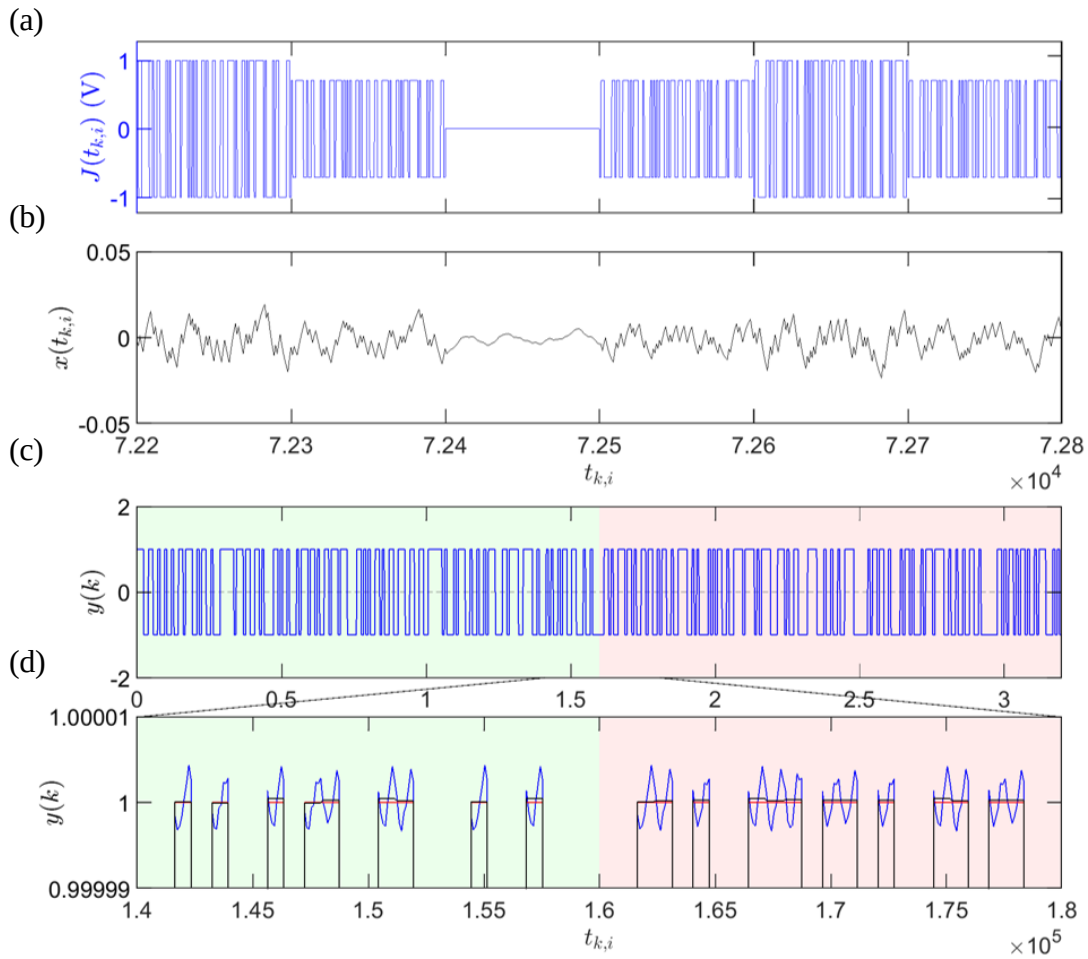


Figure 5.3: Example result of waveform discrimination using the MGO *with* delayed feedback. (a) A subsection of the input sequence $J(t)$ showing part of a masked sine wave. (b) The response of the MGO to the stimulus shown in (a) using $N = 100$ virtual nodes and a response time $T = 5\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 1.41×10^{-6} for both the training data (green background) and testing data (pink background). The error is so tiny that y is completely obscured by $\hat{y}$, so a zoomed plot of (c) is shown in (d) where variations in $\hat{y}$ can be observed. The black line represents $\langle\hat{y}\rangle$: the average value of $\hat{y}$ over each waveform period.

Optimising N and T with delayed Feedback

To find the optimum number of virtual neurons N and optimal time constant T, the waveform discrimination task was repeated for a wide range of different N-T combinations. The resulting NRMSEs and classification scores for the testing data are shown as colourmaps in Figure 5.4 (a) and (b) respectively.

The dark blue area which covers most of Figure 5.4 (a) represents low NRMSE values (< 0.05) which correspond to good performance. The NRMSE is low for all values of $T <$ 100 and for all values of $N > 40$. For smaller values of N, the NRMSE begins to increase with decreasing N. The observation of a minimum requirement for the number of virtual

nodes is consistent with Ref. [204] which reports good performance at the waveform discrimination task only when $N > 24$.

Figure 5.4 (b) shows the classifications score as a function of N and T. The classification score is 100% for all of the tested N-T combinations except those for $N \leq 4$ and $T > 10$. The difference between the trends in Figure 5.4 (a) and (b) demonstrates that even though some N-T combinations produce larger discrepancies between $\hat{y}$ and y, the classifications can still be successfully performed. This is due to output squashing which

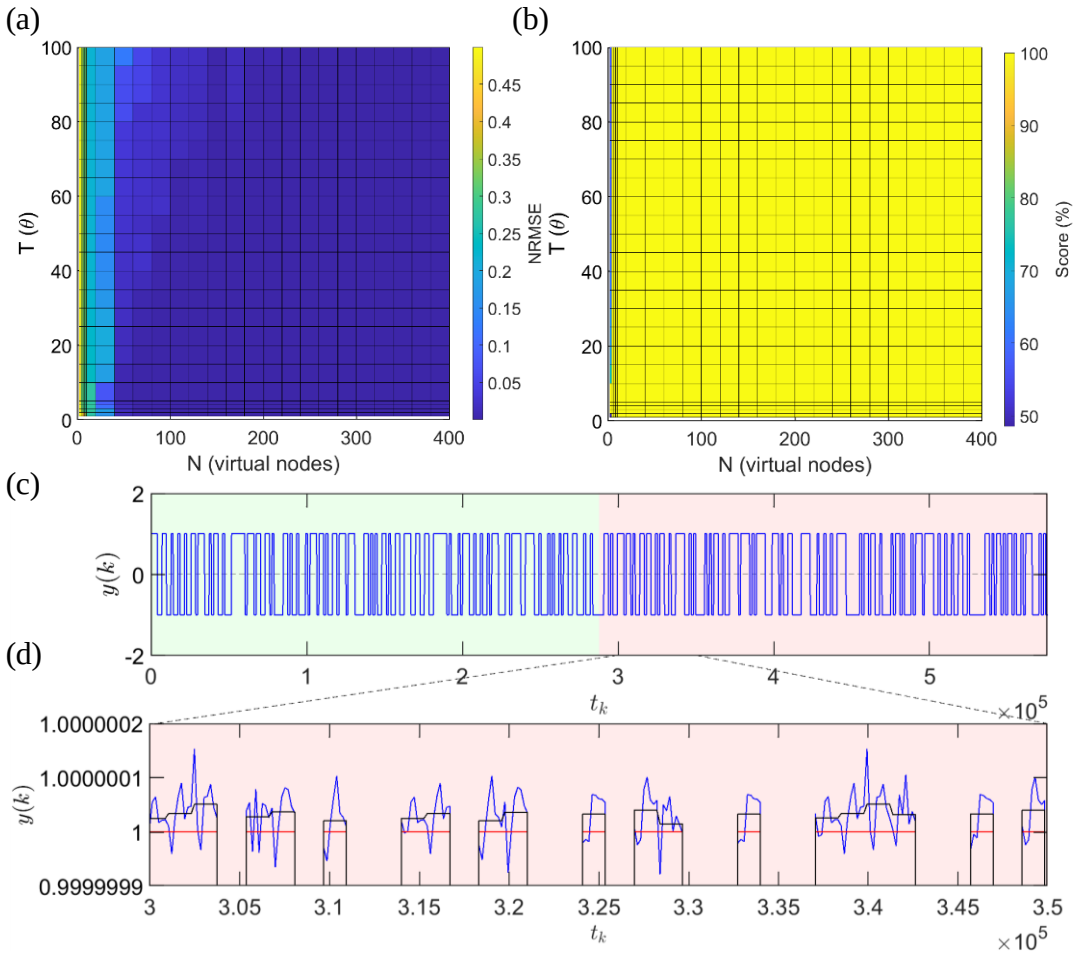


Figure 5.4: Optimising N and T for the waveform discrimination task using the MGO *with* delayed feedback. (a) Colour map representing the testing period NRMSE for different values of N and T for the waveform discrimination task using the MGO *with feedback* as the dynamical node. Lower NRMSE indicates better performance. (b) Colour map representing the number of correct classifications as a % score. (c) the best performance was observed at $N, T = 180, 1\theta$ where the virtual reservoir output $\hat{y}$ (blue) approximates the target function y (red) with a NRMSE of 3.6×10^{-8} and 3.7×10^{-8} for the training (green background) and testing (red background) periods respectively. The error is so tiny that y is completely obscured by $\hat{y}$, so a zoomed region of the plot is shown in (d) where variations in $\hat{y}$ can be observed. The black line represents $\langle \hat{y} \rangle$: the average value of $\hat{y}$ over each waveform period.

reduces the eight values in $\hat{y}$ for each waveform to a single value $\langle\hat{y}\rangle$ which is used to assign the predicted class.

The optimal performance was found for $N = 180$ and $T = 1\theta$, the result of which is shown in Figure 5.4 (c). The error is extremely small: the NRMSE is 3.6×10^{-8} and 3.7×10^{-8} for the training and testing data respectively and so a zoomed plot of Figure 5.4 (c) is shown in Figure 5.4 (d) where $\hat{y}$ (blue), y (red) and $\langle\hat{y}\rangle$ (black) can each be resolved. Note that the improvement in NRMSE necessitates an expanded vertical scale compared to Figure 5.3 (d).

Example Result without Delayed Feedback

The waveform discrimination task was also performed using the MGO *without delayed feedback*, in order to compare with the previous section and thereby characterise the role and importance of the delayed feedback component for classification tasks.

Figure 5.5 shows the same example as in Figure 5.3 ($N = 100$, $T = 5\theta$) but performed without the delayed feedback mechanism. The absence of delayed feedback is reflected in the MGO response x in Figure 5.5 (b). When the input sequence $J(t)$ in Figure 5.5 (a) is zero, x saturates to a constant value, whereas during the same period in Figure 5.3 (b), x continues to oscillate because the MGO is still receiving delayed feedback from the previous time step.

The results, shown in Figure 5.5 (c) and (d) are significantly different than in the case with delayed feedback in Figure 5.3 (c) and (d). $\hat{y}$ exhibits very large fluctuations when the delayed feedback mechanism is not included, and the corresponding NRMSE is 0.323; approximately 5 orders of magnitude greater than when delayed feedback is included. Despite this, the squashed output $\langle\hat{y}\rangle$ is well resolved for each waveform class and consequently the classification score is 100%.

Optimising N and T without Delayed Feedback

In order to find the optimal parameters, the waveform discrimination task was again repeated for a wide range of different N-T combinations. The resulting NRMSEs and classification scores for the testing data are shown as colourmaps in Figure 5.6 (a) and (b) respectively.

In the absence of delayed feedback, the NRMSE values in Figure 5.6 (a) are much higher across the entire N-T plane than in Figure 5.4 (a), signalling reduced performance. There is however an interesting new feature that emerges as a result of removing the delayed feedback mechanism: the diagonal; dark blue band which indicates optimal performance for $T \sim N\theta = \tau$. The reason for optimal performance in this region is discussed in the next section below.

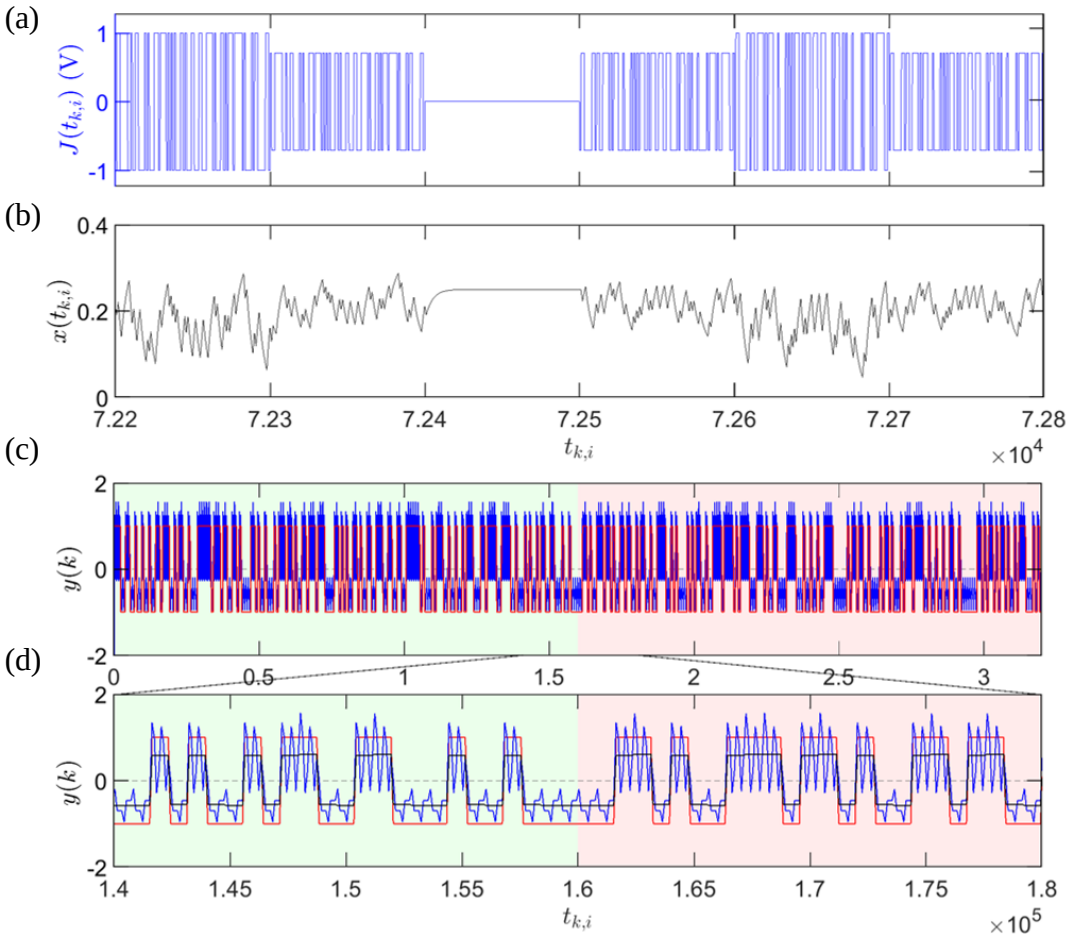


Figure 5.5: Example result of waveform discrimination using the MGO *without* delayed feedback. (a) A subsection of the input sequence $J(t)$ showing part of a masked sine wave. (b) The response of the MGO to the stimulus shown in (a) using $N = 100$ virtual nodes and a response time $T = 5\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.316 and 0.323 for the training data (green background) and testing data (pink background) respectively. (d) A zoomed region of (c) where fluctuations in $\hat{y}$ can be seen more closely. The black line represents $\langle \hat{y} \rangle$: the average value of $\hat{y}$ over each waveform period.

Again, the classification scores in Figure 5.6 (b) are largely independent of the NRMSE values with 100% of waveforms correctly classified for almost all N-T combinations.

The optimal result (lowest NRMSE) was found for $N = 8$ and $T = 10\theta$ and is shown in Figure 5.6 (c) for the full 400 waveform cycles, and a zoomed plot showing 50 waveform cycles in Figure 5.6 (d). The fluctuations in $\hat{y}$ are reduced compared to the example in Figure 5.5, which is reflected in the smaller testing NRMSE value of 0.247.

The Effect of Delayed Feedback

The inclusion of the delayed feedback mechanism clearly has an effect, not only on the NRMSE, but also on the optimal choice of N and T. In the case where delayed feedback was

included, the virtual reservoir was capable of good performance for any T, so long as there was a sufficient number ($N \geq 40$) of virtual nodes in the reservoir. However, when the feedback was removed, optimal performance was only observed for $T \sim N\theta = \tau$, indicated by the dark blue band in Figure 5.6 (a). This can be understood by considering the role of the delayed feedback mechanism, and the memory requirements of the task. The delayed feedback mechanism is responsible for emulating the recurrent connections in the virtual reservoir, which allows information injected at previous time steps (τ's) to be retained by the

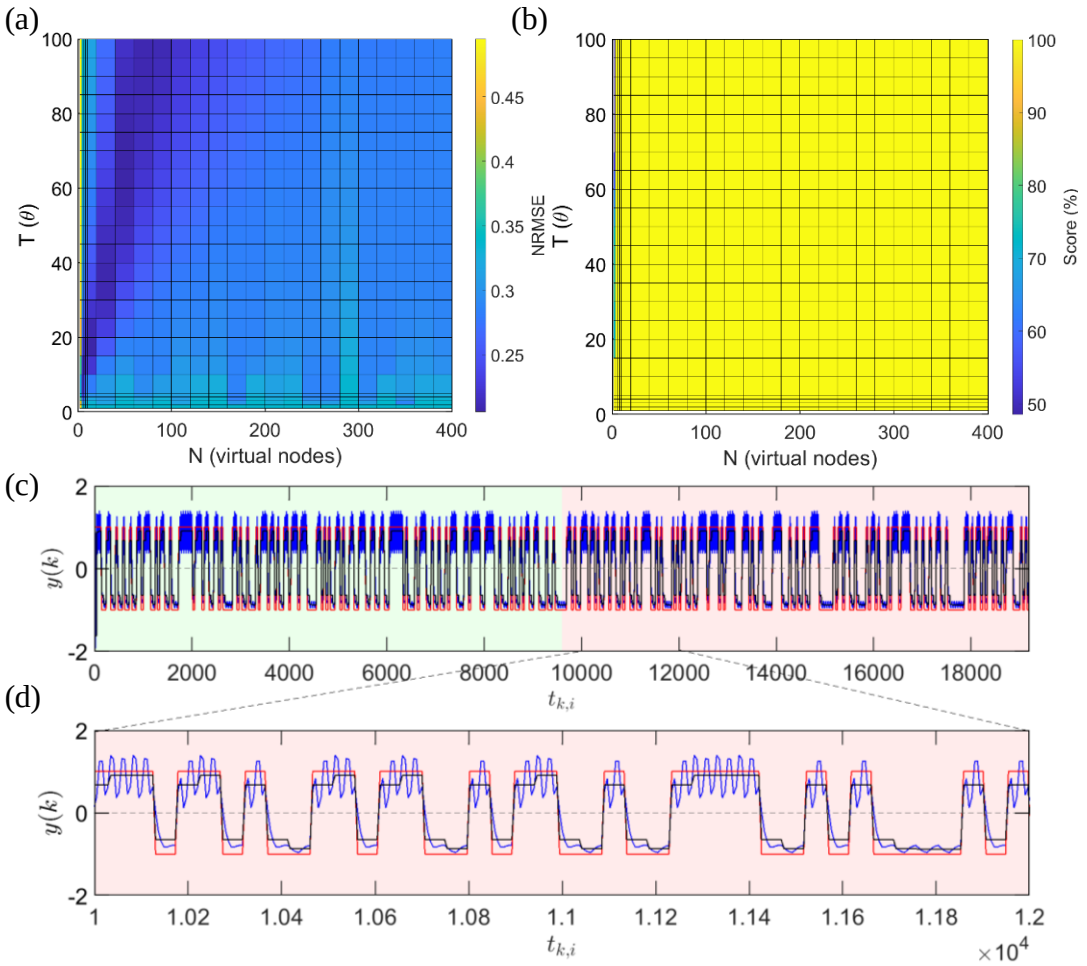


Figure 5.6: Optimising N and T for the waveform discrimination task using the MGO *without* delayed feedback. (a) Colour map representing the testing NRMSE for different values of N and T for the waveform discrimination task using the MGO *without feedback* as the dynamical node. (b) Colour map representing the classification score for different N and T. (c) The best performance was observed at $N, T = 8, 10\theta$ where the virtual reservoir output $\hat{y}$ (blue) approximates the target function y (red) with a NRMSE of 0.205 and 0.247 for the training (green background) and testing (red background) data respectively. A zoomed region of the plot is shown in (d) where variations in $\hat{y}$ can be observed. The black line represents $\langle \hat{y} \rangle$: the average value of $\hat{y}$ over each waveform period.

reservoir where it can contribute to later output signals. This is essential for a sequence classification task like waveform discrimination because the sine and square waveforms contain many input values i.e. 1 and -1 that are the same. In order to distinguish a sequence of eight sine wave values from a sequence of eight square wave values, the reservoir must always retain information from the previous time step. This means that the response of the reservoir at any time step k will depend on the k^{th} input value, as well as on the $(k-1)^{th}$ input value, and so the N-dimensional signal that is generated represents the *sequence* of input values. In the absence of the recurrence provided by the delayed feedback mechanism, a time constant $T \sim N\theta = \tau$ allows for information from the $(k-1)^{th}$ time step to be retained in the virtual reservoir at the k^{th} time step.

Of course, this is also true for values of $T \gg N$, but very large time constants have the consequence that the response to a particular input waveform may then also depend on the previous waveform. In this case, there is a different transient response for each waveform which depends on the waveforms which precede it. As the sequence of waveforms is random, this dependence leads to reduced performance. Therefore, the time constant must be large enough to facilitate recurrence, but not so much recurrence that past examples have a significant effect on the reservoir response to the present example. Hence, the optimal N-T combination corresponds to $T \sim N\theta = \tau$, where the time constant is approximately the same duration as the delay line i.e. one time step.

To further illustrate the role of recurrence in sequence classification, Figure 5.7 (inspired by Ref. [241]) shows an example of the MGO response over one time step (τ), both with and without delayed feedback. The parameters $N = 40$ and $T = 5\theta$ used in this example were chosen simply to illustrate the effect of including/excluding the delayed feedback mechanism. Figure 5.7 (a) contains a sequence of amplitude values in $I(t)$ corresponding to a sine wave followed by a square wave. There is one time step in the sine wave and four time steps in the square wave which have an amplitude of minus one. If the virtual reservoir has no recurrence, it does not retain information from previous input values, and will hence respond in the same manner to all five inputs of minus one, regardless of their place in the sequence. The MGO response to all five of the input values of minus one is shown in Figure 5.7 for the case (b) without delayed feedback and (c) with delayed feedback. The mask function $M(t)$ is shown by the grey line in each panel. In Figure 5.7 (b) where there is no feedback, the five MGO response curves, each corresponding to a different time step in the

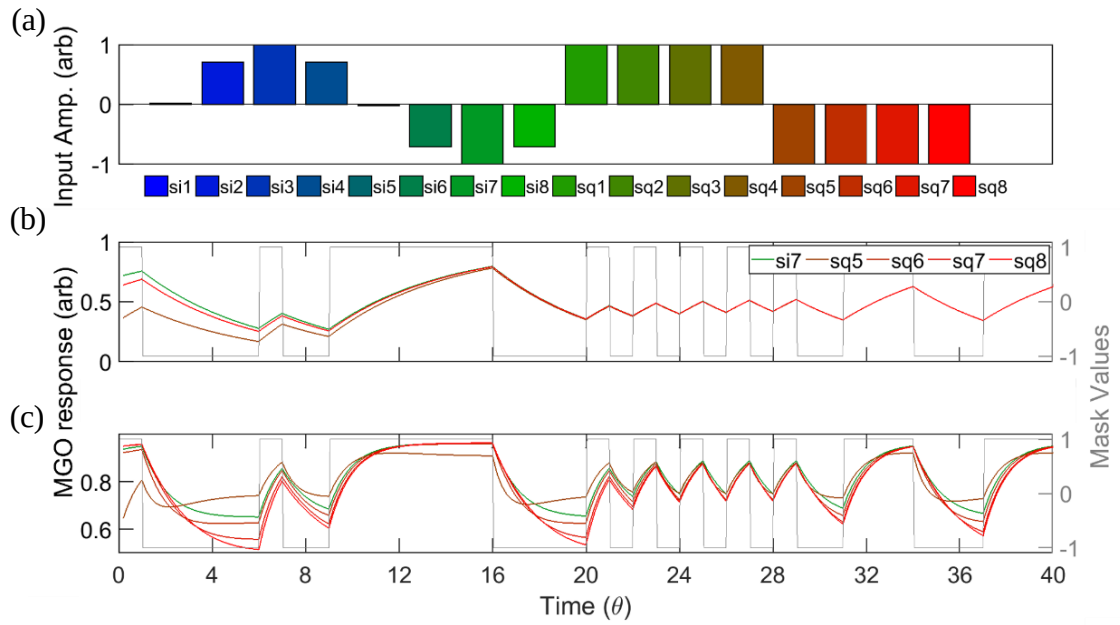


Figure 5.7: The effect of recurrence from delayed feedback. (a) A sequence of amplitude values in $I(t)$ corresponding to a sine wave followed by a square wave. Each amplitude value is applied for one time step τ, by multiplying by the mask function $M(t)$ which is shown in grey in (b, c). (b) The response of the MGO *without* feedback over different time steps which have an input amplitude of minus one. The response curve for time step si7 is only resolvable from the sq5-8 for the first few θ due to the effect of the response time $T = 5\theta$. (c) The response of the MGO *with* feedback over the same time steps shown in (b). The response curve for si7 is now resolvable from the sq5-8 curves along the entire delay line. This illustrates the increased separability of different classes when the virtual reservoir has recurrence provided by the delayed feedback mechanism. Figure inspired by Ref. [241].

sequence shown in Figure 5.7 (a), are almost identical. The exception is the short period at the beginning of the delay line where the sq6, sq7 and sq8 are identical, but sq5 and si7 are different. This is because sq5 and si7 are time steps which were preceded by amplitudes that were not minus one, and the response time of the MGO provides a fading memory of the previous time step. After $\sim 20\theta$ all five curves are indistinguishable, meaning that only some of the node states will be useful in separating the inputs. In contrast, the five MGO response curves in Figure 5.7 (c) are distinguishable along the entire delay line because of the direct recurrence provided by the delayed feedback mechanism. More importantly to the task, the si7 curve is separated from all of the sq curves. In this case, all 40 of the node states are useful in providing a higher-dimensional representation of the input sequence, and so the virtual reservoir with delayed feedback is a better classifier than the one without delayed feedback. This is consistent with the results presented in Figure 5.3 - Figure 5.6 which show that for waveform discrimination, the optimal results are obtained using virtual reservoirs with delayed feedback.

Comparison with Literature Results

The results of the waveform discrimination task presented in this section can be compared with those from the literature which are summarised in Table 1. Both with and without feedback, the optimal classification scores are 100%: all 400 waveforms in the input sequence are correctly classified. This surpasses the results from Refs. [51,210,241], although these were implementations of physical NDNs which are subject to real-world effects unlike the numerical NDN considered in this section. Refs. [146,201] report 100% classification rates, so the NRMSE can be used as an alternative performance metric. The optimal NRMSE value found here (3.7×10^{-7} with delayed feedback) exceeds both those reported in Refs. [146,201] which are 0.33 and 0.0387 respectively. It should be noted that the results reported in Refs. [146,201] are from TDRC implementations that use physical systems as the NDNs. Thus, they are subject to real-world effects like noise which are likely the reason that the reported NRMSE values are much larger than the those from the numerical MGO studied in this section. The NRMSE obtained here with no delayed feedback was 0.247 which is comparable to the result from Ref. [146] which did include a delayed feedback mechanism.

5.1.2 NARMA10 using the MGO

This section presents results of the NARMA10 task (Section 4.2.4) using the MGO as the NDN in a TDRC framework. The NARMA10 task involves using random input values $u(k)$ to predict each value in a chaotic time series y given by Eq. (44), which depends on the previous ten input and output values. The trained virtual reservoir output should be able to approximate y without explicitly knowing the relationship between $u(k)$ and y. The task was performed both with and without delayed feedback, and for a wide range of different N and T.

Example Result with Delayed Feedback

Figure 5.8 presents an example result of the NARMA10 task using the MGO *with* delayed feedback for $N = 100$ and $T = 5\theta$. Figure 5.8 (a) contains a subsection of the input sequence $J(t)$ which shows 12 time steps (i.e. 12 τ) during which 12 of the 1000 random values in the sequence $u(k)$ were fed into the virtual reservoir. Figure 5.8 (b) shows the response of the MGO (i.e. the explanatory variable $x(t)$) corresponding to the input shown in Figure 5.8 (a). Figure 5.8 (c) shows the final output of the virtual reservoir $\hat{y}$ (blue) which is the linear

combination of weighted virtual node responses given by Eq. (46). The result in Figure 5.8 (c) shows the result from 1000 input values: 500 input values were used to train the weights w_i and a further 500 input values were used to test the trained system. In Figure 5.8 (c), $\hat{y}$ is superimposed with the target function y (red), but because $\hat{y}$ is such a close approximation of y, the target function is largely obscured on this scale. Hence, a zoomed plot of Figure 5.8 (c) is shown in Figure 5.8 (d) showing the result for only 400 input values. Here $\hat{y}$ can be seen following y very closely.

The performance of the NARMA10 task is characterised by the NRMSE (calculated from $\hat{y}$ and y) which was found to be 0.053 and 0.101 for the training and testing data respectively.

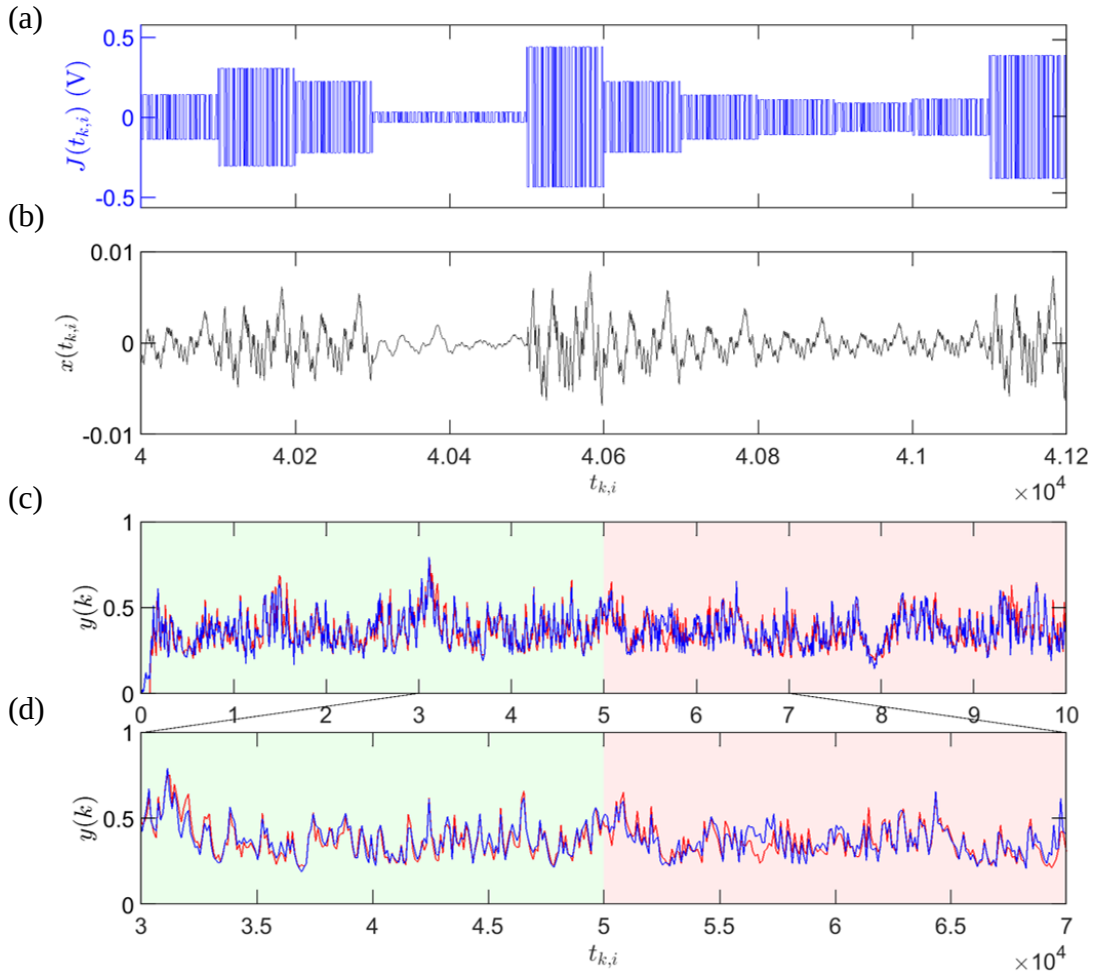


Figure 5.8: Example result of NARMA10 using the MGO *with* delayed feedback. (a) A subsection of the input sequence $J(t)$ showing 12 time steps τ corresponding to 12 of the random input values $u(k)$. (b) The response of the MGO to the stimulus shown in (a) using $N = 100$ virtual nodes and a response time $T = 5\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.053 and 0.101 for the training data (green background) and testing data (pink background) respectively. (d) A zoomed region of (c) where fluctuations in $\hat{y}$ can be seen more closely.

It is typical that the training performance is slightly better than the testing performance because the weights are adapted specifically to map the response of the reservoir to the target function *for the training data*. If the testing performance is significantly worse than the training performance, it is a sign of overfitting, which may be mitigated by regularization techniques (Section 4.3.5). Hence, small differences between the training and testing NRMSEs as observed here are of no concern.

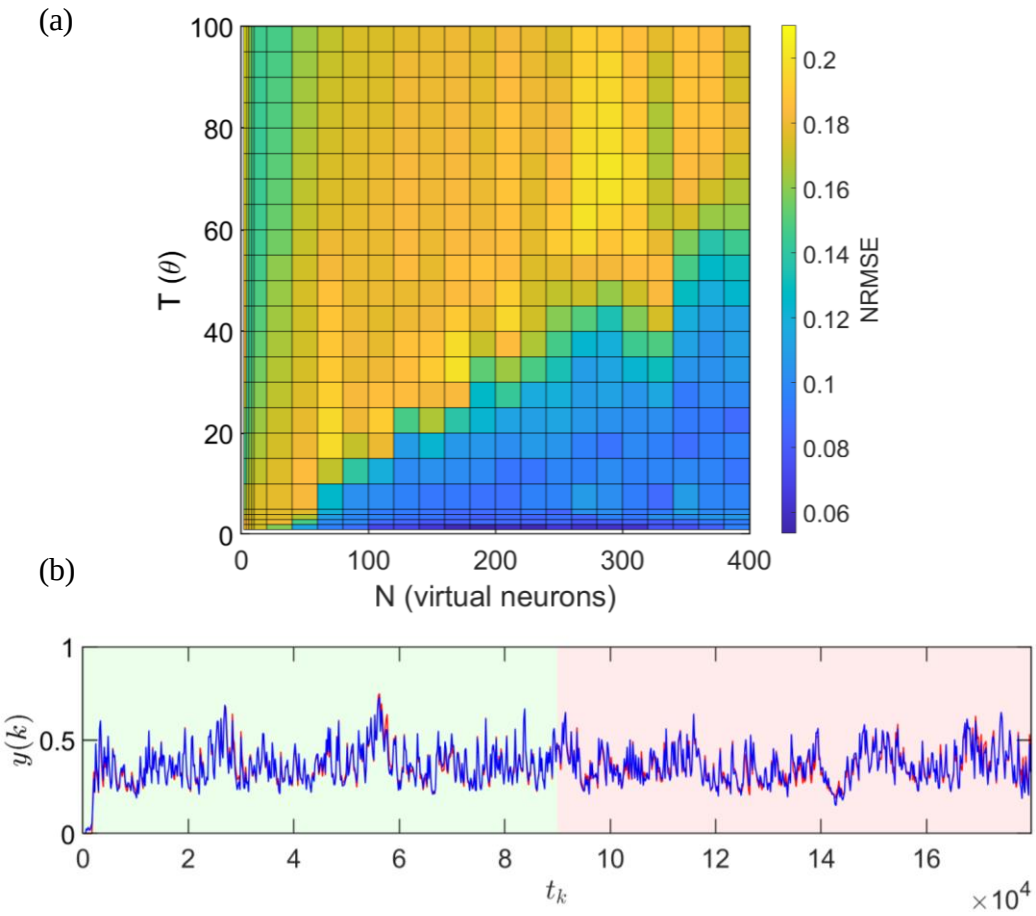


Figure 5.9: Optimising *N* and *T* for the NARMA10 task using the MGO *with* delayed feedback. (a) Colour map representing the testing period NRMSE for different values of N and T for the NARMA10 task using the MGO *with feedback* as the dynamical node. Lower NRMSE indicates better performance. (b) The result from the best performing case: $N, T = 180, 1\theta$ where the virtual reservoir output $\hat{y}$ (blue) approximates the NARMA10 target function y (red) with a NRMSE of 0.026 and 0.053 for the training (green background) and testing (red background) periods respectively.

Optimising N and T with Delayed Feedback

To find the optimum number of virtual neurons N and optimal time constant T, the NARMA10 task was repeated for a wide range of different N-T combinations. The resulting NRMSEs for the testing data are shown as a colourmap in Figure 5.9 (a). The dark blue area at the bottom of Figure 5.9 (a) represents low NRMSE values (< 0.07) which correspond to optimal performance. The NRMSE is low for values of N between 100 and 300 and when $T = 1\theta$. For larger values of T, the NRMSE begins to increase rapidly with increasing T. The observation of a short optimal time constant is consistent with Ref. [205] which reports optimal performance for the NARMA10 task when $T = 5\theta$, although curiously the optimal result found here is for $T = 1\theta$, while $T = 5\theta$ results in a NRMSE which is approximately twice as large.

Figure 5.9 (b) shows the result for the optimal N-T combination. The difference between y and $\hat{y}$ is characterised by the NRMSE which is 0.026 and 0.053 for the training and testing data respectively.

Example Result without Delayed Feedback

The NARMA10 task was also performed using the MGO *without delayed feedback*, in order to characterise the role and importance of the delayed feedback component for time series prediction tasks.

Figure 5.10 shows the same example as in Figure 5.8 ($N = 100$, $T = 5\theta$) but performed without the delayed feedback mechanism. The absence of delayed feedback is reflected in the MGO response x which is markedly different in Figure 5.10 (b) as compared with Figure 5.8 (b), despite the input sequences (including masks) being identical. The final result, shown in Figure 5.10 (c) with a zoomed plot in Figure 5.10 (d), also differ significantly from the case with delayed feedback in Figure 5.8 (c) and (d). $\hat{y}$ makes a good approximation of the high frequency oscillations in y, but does not fit the slower oscillations well. Consequently, the NRMSE is much larger than in the case with delayed feedback: 0.135 for the training data and 0.187 for the testing data. Removing the delayed feedback mechanism has thus increased the error by a factor of 3-5. Further discussion on the effect of delayed feedback for the NARMA10 task is given later in this section.

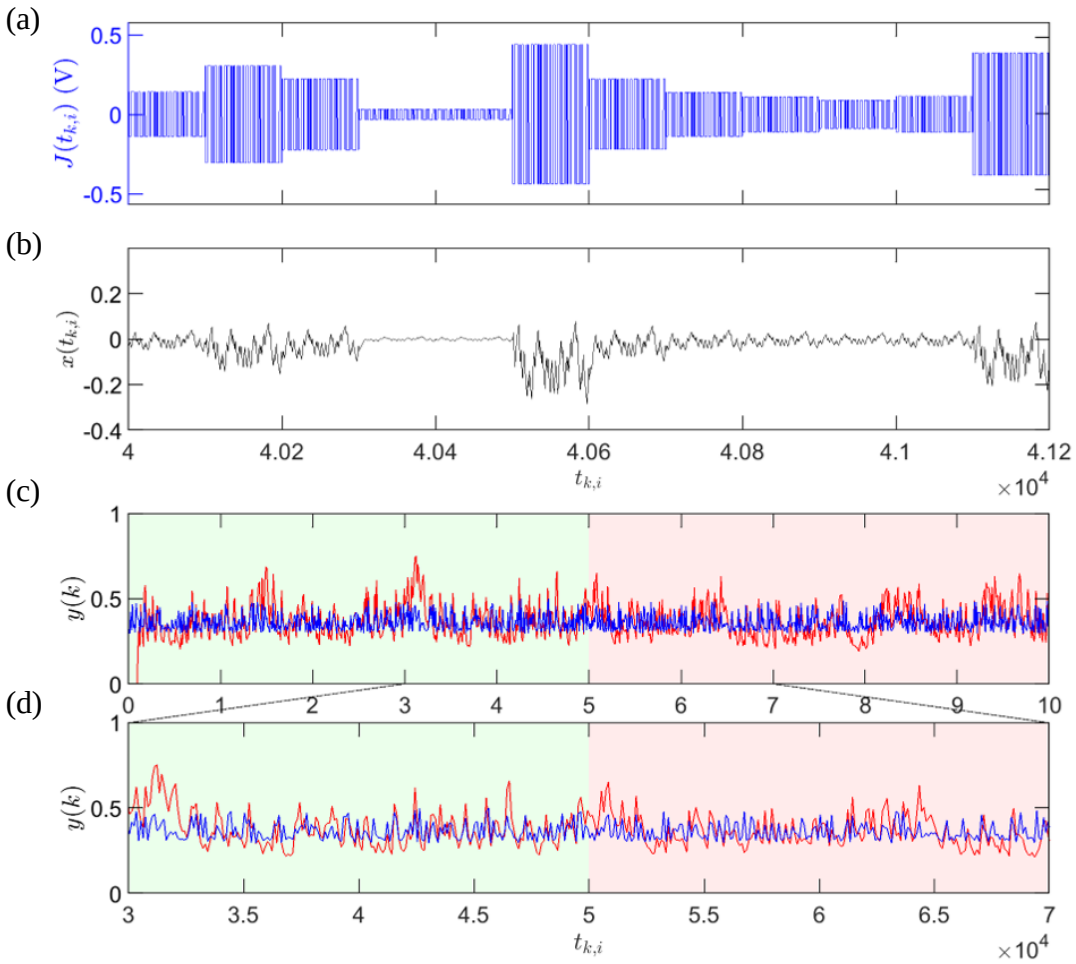


Figure 5.10: Example result of NARMA10 using the MGO *without* delayed feedback. (a) A subsection of the input sequence $J(t)$ showing 12 time steps τ corresponding to 12 of the random input values $u(k)$. (b) The response of the MGO to the stimulus shown in (a) using $N = 100$ virtual nodes and a response time $T = 5\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.135 and 0.187 for the training data (green background) and testing data (pink background) respectively. (d) A zoomed region of (c) where fluctuations in $\hat{y}$ can be seen more closely.

Optimising N and T without Delayed Feedback

The NARMA10 task was again repeated for a wide range of different N-T combinations. The resulting NRMSEs for the testing data are represented as a colourmap in Figure 5.11 (a). In the absence of delayed feedback, the NRMSE values in Figure 5.11 (a) are larger across the N-T plane as compared with Figure 5.9 (a) where delayed feedback was included. As for the waveform discrimination task in Section 5.1.1, an interesting trend has emerged as a result of removing the delayed feedback mechanism. This time the dark blue band indicates optimal performance for $T \sim 10N\theta = 10\tau$. The reason for optimal performance in this region is discussed in the next subsection below.

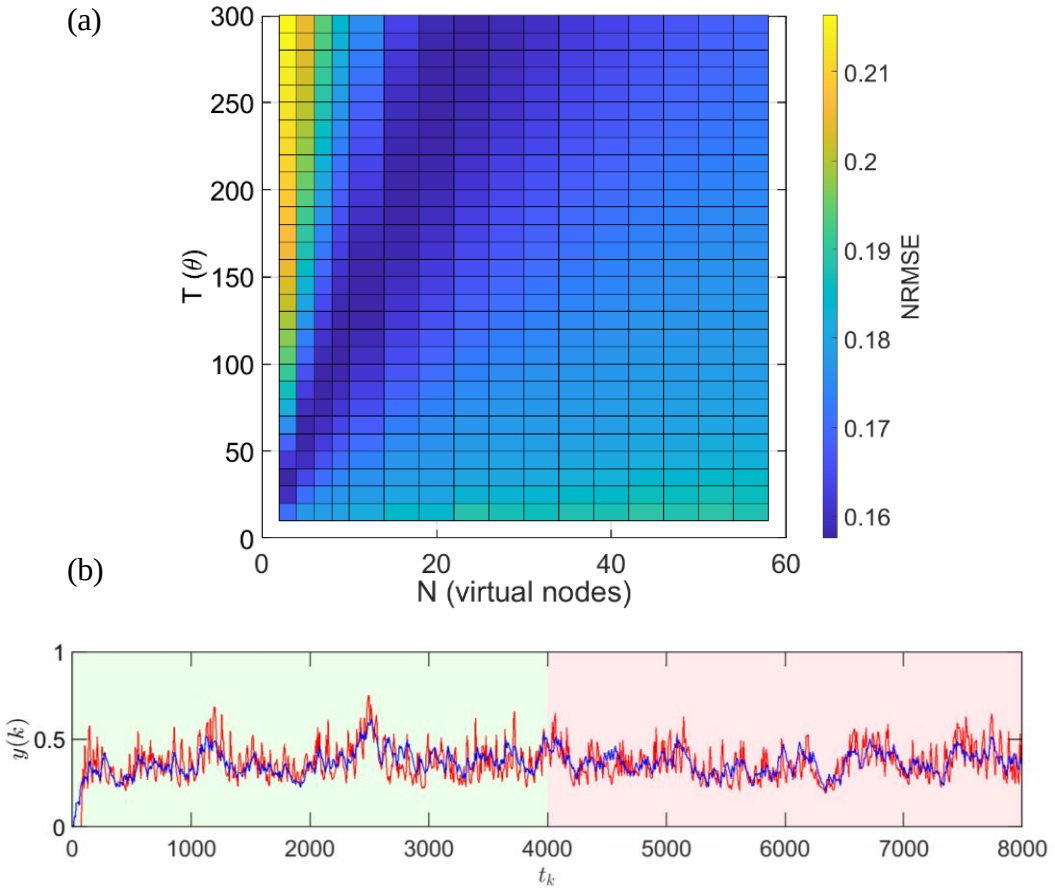


Figure 5.11: Optimising N and T for the NARMA10 task using the MGO *without* delayed feedback. (a) Colour map representing the testing period NRMSE for different values of N and T for the NARMA10 task using the MGO *without feedback* as the dynamical node. Lower NRMSE indicates better performance, with the best performance (b) being observed at $N, T = 8, 110\theta$ where the virtual reservoir output $\hat{y}$ (blue) approximates the NARMA10 target function y (red) with a NRMSE of 0.109 and 0.157 for the training (green background) and testing (red background) periods respectively.

The optimal result without delayed feedback was found for $N = 8$ and $T = 110\theta$ and is shown in Figure 5.11 (b) for the full 1000 input values. NRMSE values of 0.108 and 0.157 were found for the training and testing segments respectively. In Figure 5.10, where the time constant was $T = 5\theta$, the high frequency oscillations in y were well approximated by the virtual reservoir, but not the slow oscillations. The opposite is true for the optimal result in Figure 5.11 (b) which shows a good fit to the low frequency oscillations but not to the high frequency oscillations. This is a direct consequence of the different time constants used in each case, as will be elucidated by the following discussion regarding Figure 5.12.

The Effect of Delayed Feedback

The inclusion of the delayed feedback mechanism clearly has an effect on the optimal choice of N and T. In the case where feedback was included, the virtual reservoir was capable of good performance for a wide range of N, so long as the response time of the NDN was short. When the feedback was removed, a strong relationship between N and T emerged. This can be understood by considering the role of the delayed feedback mechanism, and the memory requirements of the task. The delayed feedback mechanism is responsible for emulating the recurrent connections in the virtual reservoir, which allows information injected at previous time steps to be retained by the reservoir where it can contribute to later output signals. This is essential for a time series prediction task like NARMA10, because each point in the target function is determined by the preceding 10 input and output values. When the delayed feedback is facilitating this recurrence, the time constant T is responsible for emulating the connectivity between virtual nodes, as discussed in Section 4.3.4.1. However, when the delayed feedback mechanism is absent, some recurrence can be provided by having a very long time constant.

If the NDN response time extends beyond $\tau = N\theta$, then the dynamical node output $x(t = k\tau + n\theta)$ has some dependence on the output at $t = (k-1)\tau + n\theta$ even if there is no delayed feedback connection directly transferring information across time steps. The caveat is that $x(k\tau + n\theta)$ then also depends on $x(k\tau + (n-i)\theta)$ where $i \in [0,\ N-1]$ i.e. the NDN states which occurred between $t = (k-1)\tau + n\theta$ and $t = k\tau + n\theta$. This is because for a long time constant, information from the previous time step is being passed along the delay line, from virtual node to virtual node, each time being altered by the state of that virtual node. So rather than each virtual node receiving its exact output from the previous time step as the delayed feedback mechanism provides, each virtual node receives a version of its previous output which has been augmented by the other virtual node states along the delay line. The dark band in Figure 5.11 (a) therefore represents the optimal point in a trade-off: if T is too small, then information is not being passed from virtual node n at $t = (k-1)\tau$ to virtual node n at $t = k\tau$ and there is no recurrent connection (the reservoir essentially becomes a feed-forward network (Section 1.2.1)) which has insufficient memory to accurately predict the NARMA10 time series. On the other hand, if T is too large, then the recurrent information being passed to virtual node n at $t = k\tau$ is strongly perturbed by the other virtual nodes on the delay line leading to a "scrambling" of the recurrent information.

The dark band in Figure 5.11 (a) indicates that when $T \sim 10N\theta = 10\tau$, the no-feedback virtual reservoir has the optimal balance of recurrence and minimal scrambling of the recurrent information. The fact that this optimal time constant corresponds to 10τ is consistent with the requirements of the NARMA10 task: the virtual reservoir needs to retain information from at least previous ten input values and reservoir states i.e. from the previous ten loops around the delay line, in order to successfully predict the next point in the sequence.

The above explanation of optimal performance at $T \sim 10N\theta = 10\tau$ (when delayed feedback is not included) is supported by the observation of the best performance at $N = 8$ virtual nodes. $N = 8$ is a very small number of virtual nodes to achieve optimal performance: the reservoir requires $N \geq 2$ to project the inputs onto a space of higher dimensionality (Section 4.3.4.1) but the recurrent information is increasingly scrambled the more virtual nodes are on the delay line. The results for the case where feedback is included gives optimal performance for $N = 180$ (Figure 5.9) because the scrambling effect is not an issue when

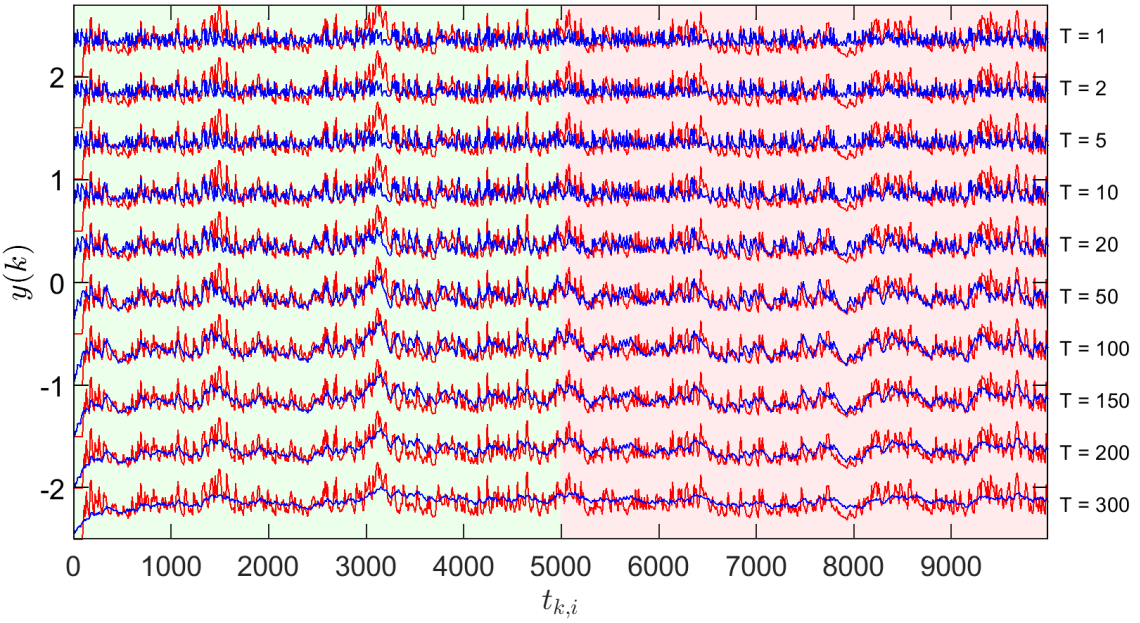


Figure 5.12: Results of the NARMA10 task for MGO *without* delayed feedback for different *T*. The NARMA10 task was repeated for $N = 10$ using a range of different time constants, given at the right of each trace in units of θ. Shorter response times result in a reservoir which can approximate the high frequency oscillations in the chaotic NARMA10 time series but fails to replicate the lower frequency oscillations. Long time constants have the opposite effect: low frequency oscillations are well approximated while high frequency oscillations are highly damped. The optimal time constant lies in between these extremes at $T \sim 10N\theta = 10\tau$, which corresponds to ten loops of the delay line i.e. ten time steps: the number of previous input and output points on which each value in the target function depends.

the time constant is short and the delayed feedback is providing the reservoir with direct recurrent connectivity.

Another sign of the trade-off between recurrence and scrambling of recurrent information (when delayed feedback is not included) can be seen in the reservoir output $\hat{y}$ i.e. the blue curve in Figure 5.11 (b). The target function y (red curve) is a chaotic time series and therefore contains a wide range of frequency components. Both high and low frequency oscillations are approximated by $\hat{y}$ when the optimal parameters are used. In Figure 5.12 $\hat{y}$ and y are plotted for $N = 10$ for a range of different time constants. Each dataset is vertically shifted for comparison, so the absolute scale of the y-axis is irrelevant. The interesting feature is in the time domain: when T is small the reservoir is capable of reproducing the high frequency oscillations in the target function, but the low frequency oscillations in $\hat{y}$ are highly damped. Conversely, for large T the low frequency oscillations are well approximated but not the high frequency components. Both of these cases result in sub-optimal performance characterised by a higher NRMSE. The lowest NRMSE, corresponding to the best performance in the absence of delayed feedback, is obtained when the amplitude of high and low frequency oscillations is balanced which only happens when the MGO response time is approximately ten time steps.

Comparison with Literature Results

The NRMSE values obtained from performing the NARMA10 task with the MGO are comparable to other results from the literature which can be found in Table 5. Here test results of 0.053 and 0.157 were obtained for cases with and without a delayed feedback mechanism. Interestingly, the result where feedback was *not* included is similar to that reported in Ref. [205] for $N = 200$ which *did* include delayed feedback; although it should be noted that Ref. [205] is an experimental result and is hence limited by noise which is not present in the numerical system studied in this section. The MGO result with delayed feedback surpasses that of Ref. [245], a numerical echo-state network approach using $N = 400$ which results in a NRMSE value of 0.099.

Ref. [205] reports a NRMSE value of 0.18 for the NARMA10 task using a simulation of their electronic circuit implementation. 0.18 is a significantly larger NRMSE (representing poorer performance) than the one obtained using the MGO in this thesis. The fine details of the numerical simulations in Ref. [205] are not clear (e.g. whether noise was included in the simulation), but the different optimal time constant of $T = 5\theta$ and the poorer performance

suggests that despite using the same model parameters, there is some difference between the simulations in Ref. [205] and the ones performed here.

A preliminary study was performed to explore the impact when synthetic white noise was added to the MGO output and the NARMA10 task was repeated (results not shown). The inclusion of white noise did increase the NRMSE but had no effect on the optimal time constant which remained at $T = \theta$. These differences are intriguing but the aim here is not to reproduce the results in Ref. [205], but rather establish a working TDRC framework that can be used as the basis of an exploration of reservoir computing using PASNs. The very low NRMSE obtained for the waveform discrimination and NARMA10 tasks (with delayed feedback) demonstrates that the framework developed performs at least as well as that discussed in the literature, and so this framework is used throughout the rest of this chapter with different NDN models.

5.2 Tunnelling Regime NDN

PASNs exhibit two distinct types of behaviour when subject to an external applied voltage. Below some threshold voltage the PASN operates in the tunnelling regime (Section 1.4.4) where the network structure is fixed and the PASN conductance (G) depends nonlinearly on the applied voltage (V). The second type of behaviour occurs when V exceeds the threshold and causes atomic scale filaments to form and rupture within the tunnel gaps throughout the network (switching regime), resulting in rich conductance switching in the form of critical avalanche dynamics (Section 3.2.3).

The MGO, which was used to produce the results in Section 5.1, is an established model for the NDN in TDRC [202]. It is therefore a useful model for exploring the TDRC parameter space and the role of the NDN, but a PASN will not respond to external inputs in the same manner as the MGO in either the tunnelling or switching regimes. Hence, this section presents a numerical NDN model of a PASN in the tunnelling regime called the Tunnelling Regime Nonlinear Dynamical Node (TRNDN).

5.2.1 TRNDN Model

In the tunnelling regime, a PASN could be used as a NDN by encoding input signals $J(t)$ as applied voltages, and the conductance could be used as the explanatory variable x. To explore

this possibility numerically, the TRNDN model incorporated the experimentally observed nonlinear relationship between G and V and a modified version of Eq. (51). The general form of which is

$$\frac{dx}{dt} = \frac{1}{T}[\zeta(x_\tau + \kappa J(t)) - x(t)] \tag{53}$$

where T is the response time constant, x_τ denotes the delayed dynamical variable x at time $t - \tau$, κ is a positive constant (set to 0.5 here), and ζ represents any nonlinear transfer function. Eq. (53) features the vital components of a NDN: the factor of $1/T$ provides the connectivity between virtual nodes. Just as in the MGO (Figure 4.7), recurrence is provided by the delay-dynamical variable x_τ and the nonlinearity comes from the function ζ.

In theory, Eq. (53) can be used to explore a wide range of nonlinearities simply by changing the function ζ. However, the aim of this chapter is to implement a PASN as a physical NDN, so ζ is modelled on the nonlinear relationship between G and V observed experimentally in PASNs in the tunnelling regime. The aim of this section is to demonstrate that the G-V nonlinearity of a PASN in the tunnelling regime could be used to implement an experimental PASN-based NDN for TDRC.

Tunnelling Regime Nonlinearity

The tunnelling nonlinearity was measured by applying voltage sweeps to a PASN and measuring G. Plotting G as a function of V yields the G-V characteristic shown in Figure 5.13, which is representative of most PASN devices. The Kaiser model [101] (Section 1.4.4) which is given by Eq. (1) was used to fit (red line) the measured data (blue symbols) using standard linear regression techniques which produced the parameters: $g_0 = 1.78$, $h_0 = 0.56$ and $V_0 = 4.02$. With these parameters from the experimental data, Eq. (1) was used as the nonlinear function ζ. Explicitly,

$$\zeta(V) = \frac{1.78 \exp(V/4.02)}{1 + 0.56[\exp(V/4.02) - 1]} \tag{54}$$

where $V = x_\tau + \kappa J(t)$. This is called V because it represents the actual voltage which would be applied to a PASN in an experimental implementation.

Example of TRNDN Response

Figure 5.14 (a) gives an example of the numerical TRNDN response ($G(t)$) to input signals $V(t)$. A time constant of $T = 5\theta$ was used which represents the response time of a real PASN

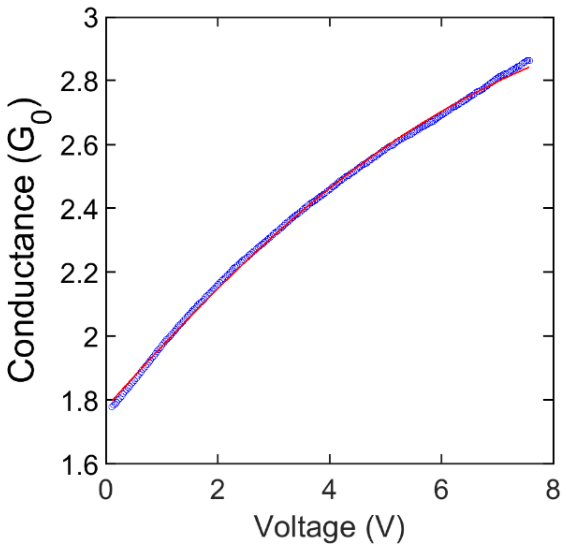


Figure 5.13: Nonlinear G-V characteristic of a PASN. The conductance measured as a function of voltage (blue) and the corresponding fit using the Kaiser model [101] (red). Here, the fitted curve is used as the nonlinear function ζ for the TRNDN for TDRC. The fitted parameters are $g_0 = 1.78$, $h_0 = 0.56$ and $V_0 = 4.02$.

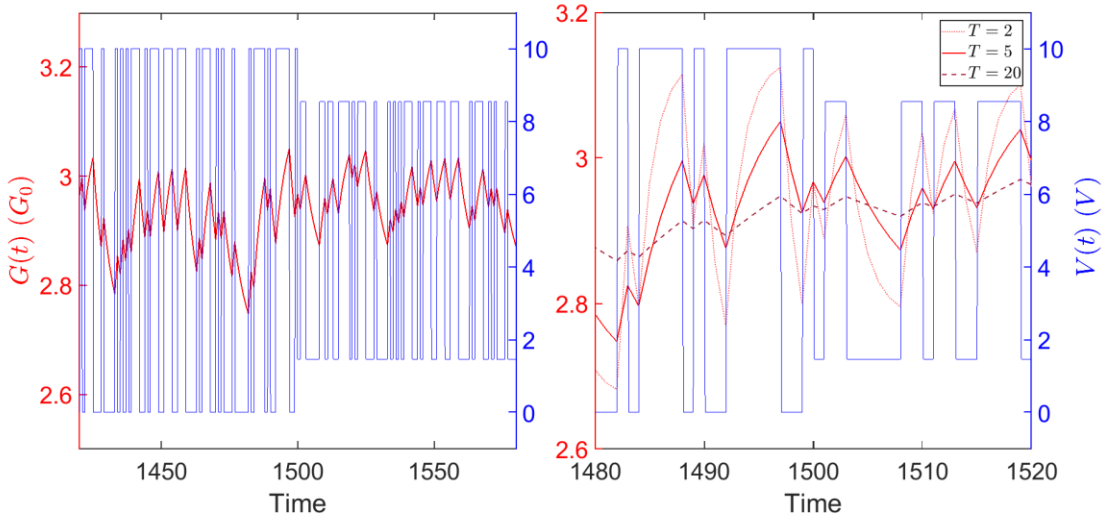


Figure 5.14: TRNDN Response. Left: The conductance response $G(t)$ of the numerical TRNDN to input signals $J(t)$ which are encoded as applied voltages $V(t)$. A time constant of $T = 5\theta$ was used. Right: A zoomed region of the left panel also showing the response of the TRNDN using two other time constants: $T = 2\theta$ (pink, dotted) and $T = 20\theta$ (maroon, dashed). When performing tasks, $G(t)$ is used as the explanatory variable x.

device. The TRNDN response exhibits qualitative similarity to the MGO response shown in Figure 5.2 (d). Figure 5.14 (b) shows a zoomed region of Figure 5.14 (a) with TRNDN response curves for three different time constants: $T = 2\theta$, $T = 5\theta$ and $T = 20\theta$. Changing the TRNDN time constant T has a similar effect as in the case of the MGO, illustrated by the comparison of Figure 5.14 (b) and Figure 4.7 (a). Thus, changing the TRNDN time constant also alters the connectivity between the virtual nodes in a similar way to the time constant in the MGO (see Figure 4.7 (b) and (c) and the related discussion in Section 4.3.4).

5.2.2 Results from the TRNDN Model

The numerical TRNDN model was used in the TDRC framework to perform the waveform discrimination and NARMA10 tasks. x is sampled N times during each time step τ, providing N virtual node states. To exploit the nonlinearity of the $G(V)$ relationship (Eq. (54)), the sequence of input waveforms for the waveform discrimination task was scaled to have an amplitude of $\pm$5V with an offset of 5V, such that the input waveforms have values between 0 and 10 V. For the NARMA10 task, the sequence of random input values $u(k) \in [0, 0.5]$ was scaled by a factor of 10 before being offset by 5V so that the input voltages have values between 0 and 10 V after masking.

The TRNDN was used to perform each task for several different cases (e.g. with/without delayed feedback etc). Since the discussion of the results for each case is quite lengthy and the cases are also superficially similar to one another, to aid readability the detailed results are presented in Appendix A and a summary of the main findings is given here.

Appendix A.1 contains the results of the waveform discrimination task. The case with delayed feedback is presented in Appendix A.1.1, and the case without delayed feedback is presented in Appendix A.1.2. It was found that the TRNDN with delayed feedback performs the waveform discrimination task extremely well, yielding a classification score of 100% and NRMSE of 6.97×10^{-7}; a similar result to that obtained using the MGO (NRMSE of 3.7×10^{-7}). When the delayed feedback mechanism was not included, the NRMSE increased to 0.202 but the classification score remained at 100%. This result is slightly better than the optimal result from the MGO without delayed feedback (NRMSE of 0.247). In both cases the trend across the N-T plane was similar to the MGO: with delayed feedback the optimal time constant was short and without delayed feedback the optimal time constant was much longer. This trend is consistent with the principle that long time constants can provide the virtual

reservoir with some recurrence and can therefore act as a substitute (albeit an inferior one) for the delayed feedback mechanism. Together these results from the numerical TRNDN are a promising sign that the implementation of a *physical* TRNDN (i.e. using a real PASN) could be used to perform classification tasks, even without a real-time delayed feedback mechanism.

Appendix A.2 contains the results of the NARMA10 task. The case with delayed feedback is presented in Appendix A.2.1. Unlike the waveform discrimination task where the TRNDN behaviour was similar to the MGO, the TRNDN did not perform well at the NARMA10 task when delayed feedback was included. The performance is characterised by a NRMSE of 0.176 (approximately 3 times larger than that of the MGO). In addition, the N-T parameter space was also different, showing optimal performance for small N where there was only weak dependence on T. It is possible that the TRNDN may be further optimised for the NARMA10 task by adapting the delayed feedback strength (κ in Eq. (53)).

Appendix A.2.2 presents the results of the NARMA10 task when delayed feedback was not included. Remarkably, the optimal result (NRMSE of 0.157) is better than the case with delayed feedback (NRMSE of 0.176); usually the inclusion of delayed feedback leads to significant improvements in performance. Although, this may simply be because the TRNDN with delayed feedback performs worse than expected at NARMA10. The trend across the N-T plane is very similar to the MGO case showing optimal performance for $T \sim 10N\theta = 10\tau$. Again, this result can be explained by the ability of long time constants to provide indirect recurrence in the absence of delayed feedback. As discussed in Section 5.1.2 for the MGO, the observation that optimal time constants are $\sim$10 time steps reflects the fact that each value in the NARMA10 target function depends on the ten previous values. The NARMA10 results obtained from the TRNDN are comparable to that reported in Ref. [205], and are a promising sign that real PASNs may be able to perform time-series prediction tasks.

Summary

The TRNDN model explored in this section provides valuable insight into how a PASN might be implemented as a physical NDN for TDRC. TDRC could be performed experimentally by encoding input signals in the applied voltage and measuring conductance as the explanatory variable *in the tunnelling regime*. The time constant could be applied by using a low pass filter [247], or by exploiting the natural time-scale of the device [204].

From the results in this section one can deduce that for optimal performance at the waveform discrimination task, the delayed feedback component should ideally be included. In this case, the PASN should perform well for $N \geq 50$ and $2\theta \leq T \leq \tau/2$. For the NARMA10 task, optimal performance should be expected for small N (≤ 60) and any time constant $T \geq 2\theta$, although the PASN may not perform well at NARMA10 with delayed feedback included.

If delayed feedback were not included, then the optimal choice of N and T would be different: for the waveform discrimination task, optimal performance should be observed for N-T combinations that lie along the dark band in Figure A.4 (a) at $T \sim N$ – though this choice may have little impact on the number of correct classifications if the virtual reservoir is capable of performing the task. A physical TRNDN should provide good results for the NARMA10 task without delayed feedback if $T \sim 10\tau$.

Experimental results using a *physical* TRNDN are described in Section 6.1.

5.3 Switching Regime NDN

Section 5.2 modelled the use of a PASN operating in the tunnelling regime and showed that the tunnelling regime could be used to perform classification/prediction tasks. This section aims to model a PASN in the switching regime. At this point the reader is advised in advance that the model was unable to achieve successful classification or successful performance of the NARMA10 task. Some readers may therefore wish to jump straight to the conclusions and explanations for this lack of success, in Section 5.3.3.

In Chapter 3 it was demonstrated that in the switching regime, PASNs exhibit sequences of highly correlated switching events characterised by power law distributed inter-event intervals and slowly decaying autocorrelation functions. It was also shown that the PASN is a critical system which displays avalanches of switching activity with power law distributed sizes and durations. This is interesting for neuromorphic applications because it has been demonstrated that criticality in the brain is responsible for optimising computational performance, information transfer and memory storage [13,33-36]. Hence, the aim is to exploit the features of criticality and correlated switching activity of PASNs to computational advantage.

This might be achieved by embedding a PASN as the NDN within a TDRC framework. If the input signals are encoded as applied voltage, and if that applied voltage exceeds the threshold for switching activity, the switching rate may be used as the explanatory variable x. It has been shown that the switching rate increases nonlinearly with applied voltage in Ref. [15], so in a TDRC framework, the switching rate should be able to facilitate the required higher-dimensional transformation of the input voltage signals. Given that the switching rate is the quantity which exhibits avalanches and features of temporal correlation, there should be some inherent memory of past inputs because the switching activity of the network always depends on a combination of present input and previous network states. Hence, the two key requirements of a NDN, nonlinearity and memory, should be fulfilled by the PASN switching rate. TDRC using the switching rate is implemented using a numerical model called the Switching Regime Nonlinear Dynamical Node (SRNDN).

5.3.1 SRNDN Model

The SRNDN was designed to mimic the experimental switching rate and the statistical properties which were identified during the analysis of avalanches and criticality in Chapter 3. The aim was to produce a generative model which produces a switching rate value for each time step, based on the present input and past outputs. When given a constant input signal (equivalent to a DC voltage), the model should produce a switching rate as a function of time, which when subjected to avalanche analysis yields statistical distributions of avalanche properties which are similar to those observed in the experimental data (Section 3.2.3).

Experimental Switching Rates

The switching rate of a PASN is found by counting the number of switching events as a function of time. The average inter-event interval (IEI) is typically used as the bin size because it is the standard used in the avalanche analysis in Section 3.1.2. Switching rate histograms can then be constructed which show the number of bins that contain each value in the switching rate. Figure 5.15 shows a typical example of (a) the experimentally measured switching rate, and (c) the corresponding switching rate histogram (blue). The switching rate is bursty, featuring highly active periods followed by periods of quiescence, and the histogram of switching rate values follow power law decay with an exponential cut off in the

tail. The SRNDN model should therefore produce a synthetic switching rate which exhibits these same properties.

SRNDN Model Description

The SRNDN draws each switching rate value from a power law cumulative distribution function (CDF), which is given by

$$CDF(x) = Ax^{\frac{1}{1-\rho}-1} \tag{55}$$

where x represents the synthetic switching rate, A is a normalization factor and ρ is the power law exponent. By using a different random number r_i at each time step, the power law distributed switching rate value x_{PL} is extracted using

$$(x_{PL})_i = (1 - r_i)^{\frac{1}{1-\rho}-1} - 1 \tag{56}$$

where $r_i \in [0, 1]$. The -1 term is included in Eq. (56) (and subsequently Eqs. (57) and (58)) to account for the number of zeros in the switching rate. Figure 5.15 (c) shows that the number of zeros in the experimental switching rate (blue) is consistent with the power law distribution and subtracting 1 from each value $(x_{PL})_i$ allows the synthetic switching rate to contain a similar proportion of zeros.

Although the histogram of x_{PL} values follows a power law, the sequence does not feature bursts because each value is selected randomly with no correlation between them. To induce burstiness in the synthesised switching rate another random number $g_i \in [0, 1]$ is therefore used. At each time step, there is equal probability that the rate should increase at the next time step ($g_i > 0.5$), and that it should decrease in the next time step ($g_i < 0.5$).[15] If the rate is increasing and the preceding rate value x_{i-1} is non-zero, then the random number r_i used to draw x_{PL} from Eq. (56) is scaled in such a way that $x_{PL} > x_{i-1}$. Specifically, $r_i \rightarrow (1 - r_{i-1})r_i + r_{i-1}$ such that Eq. (56) becomes

$$(x_{PL})_i = (1 - [(1 - r_{i-1})r_i + r_{i-1}])^{\frac{1}{1-\rho}-1} - 1 \tag{57}$$

thus, guaranteeing that the next switching rate value will be drawn from the same power law, but will be greater than the preceding value. Conversely, if the switching rate is decreasing

[15] The justification for this approach is simply that the SRNDN model produces a synthetic switching rate which is a faithful representation of the measured switching rates of PASNs (compare the green and blue plots in Figure 5.15) which features critical avalanche dynamics (compare Figure 5.17 with the results in Section 3.2.3).

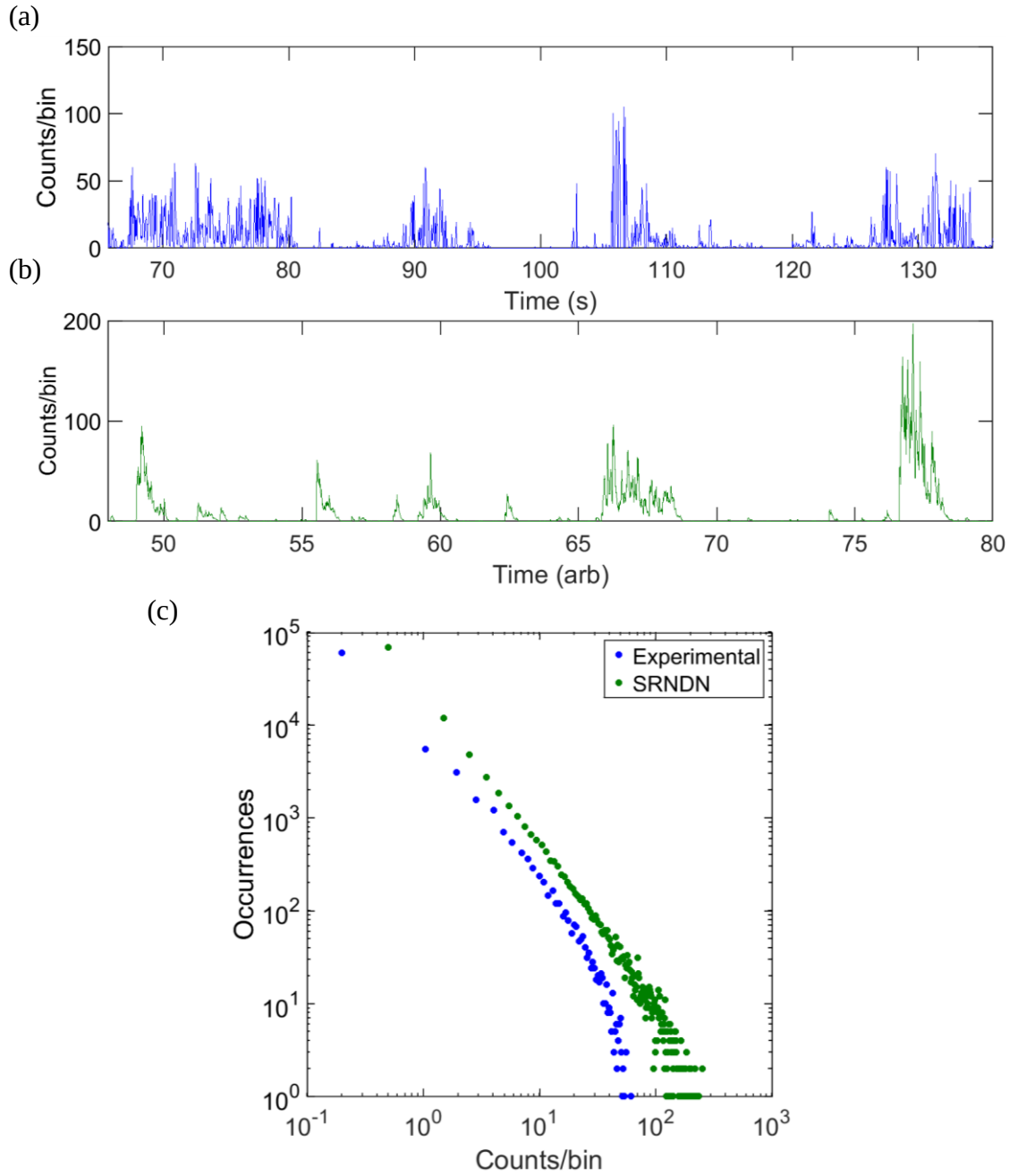


Figure 5.15: Comparison of experimental and SRNDN switching rates and switching rate histograms. (a) Subsection of a typical switching rate as a function of time measured from a PASN while applying 6V DC across the two electrodes. (b) Synthetic switching rate as a function of time from the SRNDN. The experimental data and the model data are qualitatively similar, both featuring characteristic bursts of activity. (c) The switching rate histograms from the PASN (blue) and from the SRNDN (green). The distributions are qualitatively and quantitively similar, both exhibiting power law decay with an exponential cut off and similar slopes on the logarithmic axes. Note that the largest number of occurrences for each distribution corresponds to rate values of zero. Usually on logarithmic axes, such values could not be seen, but the x-axis is plotted as bin *centres* to show that the number of zeros in the switching rate is also consistent with the power law distribution.

and the preceding rate value is non-zero, then $r_i \rightarrow (r_{i-1})r_i$ and Eq. (56) becomes

$$(x_{PL})_i = (1 - (r_{i-1})r_i])^{\frac{1}{1-\rho}-1} - 1 \tag{58}$$

which always draws a switching rate value which is smaller than the preceding switching rate value, but which belongs to the same power law. This makes the switching rate value x_i dependent on the preceding switching rate value x_{i-1} if $x_{i-1} > 0$. Finally, to add correlation between switching rate values on a longer time scale, a response time T is introduced, essentially acting as a low pass filter. The switching rate value for time step i is then given by

$$x_i = \frac{1}{T}\left(\frac{(x_{PL})_i + x_{i-N}}{2} - x_{i-1}\right) \tag{59}$$

if delayed feedback is included, and by

$$x_i = \frac{1}{T}((x_{PL})_i - x_{i-1}) \tag{60}$$

if delayed feedback is not included (N is the number of virtual nodes in the delay line). The resulting switching rate and switching rate histogram are shown in Figure 5.15 (b) and (c) (green). The switching rate is bursty and features the characteristic active periods followed by periods of quiescence, and the histogram of switching rate values follows power law decay with near identical slope to the experimental distribution.

The final requirement of the SRNDN is that it must produce a nonlinear transformation of the input voltage signals J_i i.e. the switching rate x must scale nonlinearly with applied voltage. This would be true if $x_i(J_i) = f(J) \cdot x_i$ where $f(J)$ is some nonlinear function. It was established in Ref. [15] that the PASN switching rate is a nonlinear function of voltage. This is exemplified by Figure 5.16 (a) which shows the average switching rate of a PASN in response to voltage pulses of different duration and amplitude. The relationship between the applied voltage and the average switching rate is approximately sigmoidal. The sigmoid function described by Eq. (62) and shown in Figure 5.16 (b) is therefore used to couple the switching rate values produced by Eq. (59) to the applied voltage.

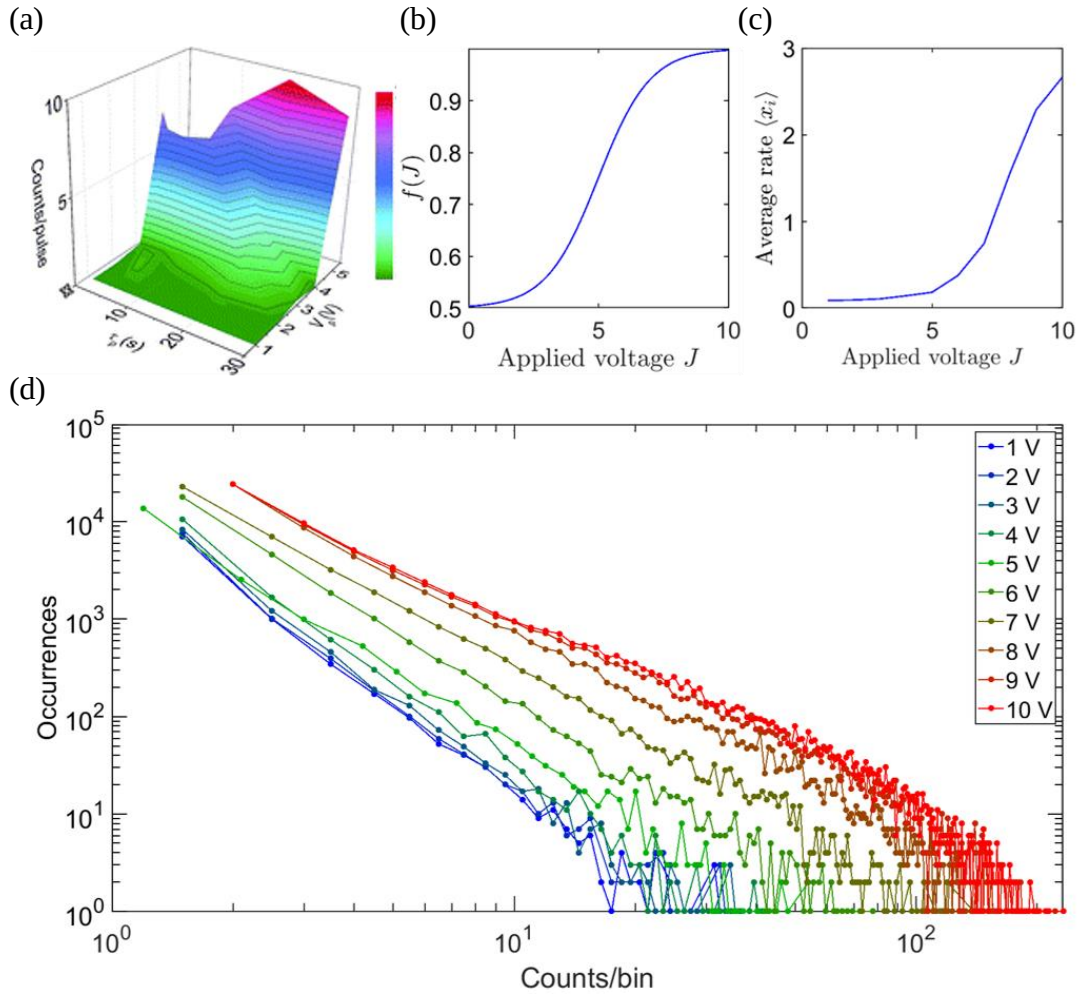


Figure 5.16: Voltage dependence of the SRNDN model. (a) The experimentally measured average PASN switching rate as a function of applied voltage and pulse duration, taken from Ref. [15]. The rate can be approximated by a sigmoidal function of voltage. (b) The sigmoid function $f(J)$ which is used to couple the SRNDN switching rate to the applied voltage J given by Eq. (62). (c) The resulting average switching rate produced by the SRNDN as a function of applied voltage. The trend matches that of the experimental data shown in (a). (d) The switching rate histograms for different applied voltages. The area under each curve represents the total number of switching events which increases with applied voltage. The heavy tail extends further with increasing voltage indicating that there are more large rate values indicating increased switching activity at higher voltage, consistent with experimental observations.

Explicitly, the final switching rate value at each time step is given by[16]

$$x_i \to f(J) \cdot x_i \tag{61}$$

where

$$f(J) = \frac{1}{2}\left(1 + \left[\frac{e^{J_i}}{e^{J_i} + 1}\right]\right) \tag{62}$$

such that

$$x_i \to \frac{x_i}{2}\left(1 + \left[\frac{e^{J_i}}{e^{J_i} + 1}\right]\right) \tag{63}$$

Because the switching rate represents a number of discrete events in a time bin, each value x_i is rounded to the nearest integer.

The average switching rate produced by the SRNDN, shown in Figure 5.16 (c), increases nonlinearly with applied voltage and is a good qualitative approximation of the experimentally observed voltage dependence (for fixed pulse duration) shown in Figure 5.16 (a). The switching rate histograms for different voltages in the range 1-10 V are shown in Figure 5.16 (d). Increasing the applied voltage causes the heavy tail of the switching rate histogram to extend further, thus emulating the increased switching activity observed in PASNs at larger applied voltages.

In summary, the SRNDN is a numerical model which produces a sequence of switching rate values which are power law distributed. The switching rate as a function of time contains correlated avalanches of activity, consistent with the demonstration of criticality given in Section 3.2 and providing some memory, which is a crucial requirement for a NDN. The average switching rate increases nonlinearly with applied voltage as required to successfully project input signals onto a higher dimensional space. Thus, the SRNDN is both a faithful representation of real PASN switching activity (see Figure 5.15) and features the vital components of an NDN for TDRC.

SRNDN Critical Avalanche Dynamics

To demonstrate that the synthetic switching rate generated by the SRNDN is a good approximation of the critical avalanche dynamics observed in real PASNs, the same

[16] The arrows in Eqs. (61) and (63) represent a variable replacement. For example, Eq. (61) says that the value of x_i produced by either Eq. (59) or (60) is going to be replaced by the value x_i multiplied by the sigmoid function f *without changing the name of the variable*. This notation was used to reduce the number of different variable names which are only being used for explanatory purposes.

avalanche analysis used in Section 3.2.3 was performed on data from the SRNDN model. The results of this analysis are shown in Figure 5.17. The avalanche size and duration distributions (Figure 5.17 (a) and (b)) follow the expected power law scaling functions given by Eqs. (32) and (33) respectively, while the average size given duration (c) follows the power law in Eq. (36). The mean temporal profiles of avalanches shown in Figure 5.17 (d) collapse onto a single universal scaling function in Figure 5.17 (e) in accordance with the unified theory of critical systems [171]. The three independent measures of the critical exponent $1/\sigma\nu z$ (see figure caption) agree within their uncertainties with those obtained from experimental data in Section 3.2.3, signalling that the switching rate produced by the SRNDN

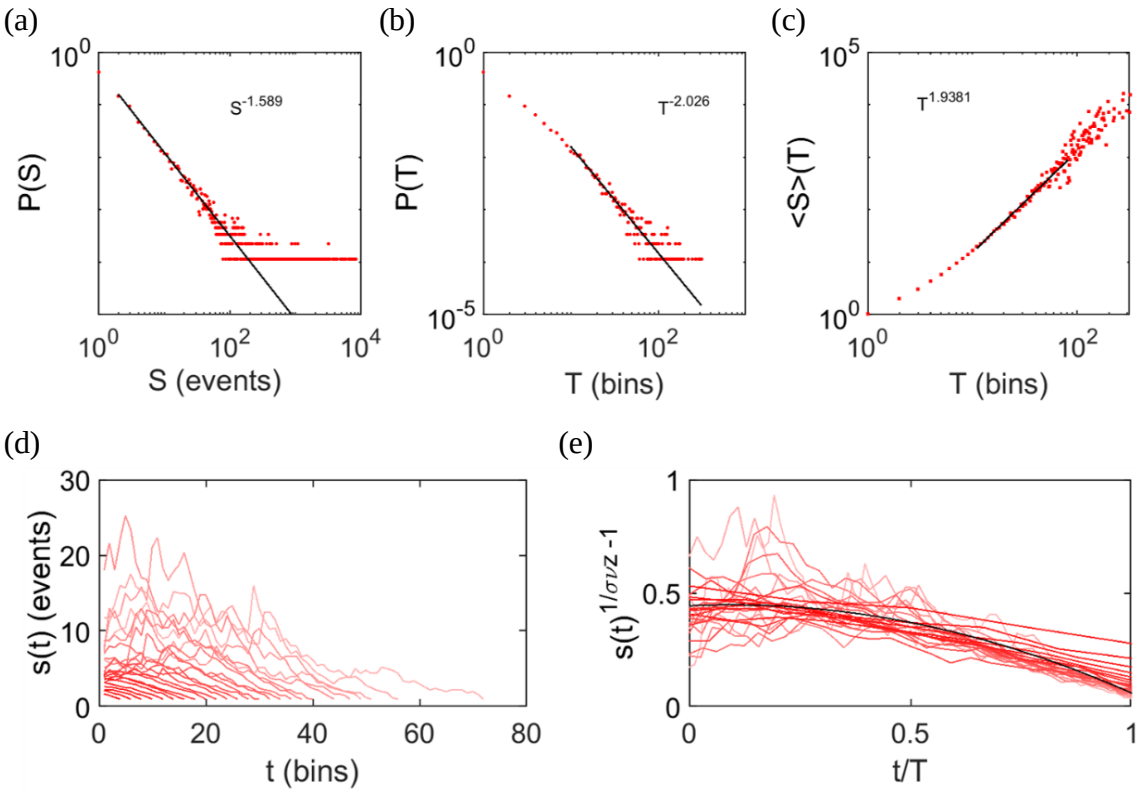


Figure 5.17: Avalanche analysis of the SRNDN model. (a) Avalanche size PDF and (b) avalanche duration PDF follow the power laws given by Eqs. (32) and (33) respectively. The critical exponents were found by MLE to be $\tau = 1.6 \pm 0.1$ and $\alpha = 2.0 \pm 0.1$. The crackling relation (Eq. (41)) can be used to find the critical exponent $1/\sigma\nu z = 1.7 \pm 0.2$ (c) The average size of avalanches scale as a power law with their duration, in accordance with Eq. (36). The critical exponent was found to be $1/\sigma\nu z = 1.94 \pm 0.03$. (d) The average temporal profiles of avalanches of different durations collapse (c) onto a universal scaling function. This operation yields another independent measure of the critical exponent $1/\sigma\nu z$ which was found to be 1.80 ± 0.06. The three measures of the critical exponent $1/\sigma\nu z$ are close to agreement, showing that the SRNDN switching rate is representative of a critical system. The values of $1/\sigma\nu z$ found from PASN avalanche analysis were similar ($1/\sigma\nu z \approx 1.7$) indicating that the two systems belong to the same universality class [171].

model is representative of the same critical avalanche dynamics as those observed in real PASNs. The SRNDN parameters used to generate the analysed data were $T = 5$ and $\rho = 2.7$, which were found by trial and error to produce a good approximation of the experimental data. $\rho = 2.7$ is henceforth adopted as a fixed parameter, while the response time T is left as a free parameter to explore the effect of memory on the SRNDN virtual reservoir performance.

5.3.2 Results from the SRNDN Model

The numerical model of the SRNDN developed in the previous section was used in the TDRC framework to perform the waveform discrimination and NARMA10 tasks. The explanatory variable x is the synthetic switching rate which is sampled N times during each time step, making up the states of the virtual nodes. To exploit the nonlinearity of the voltage-switching rate relationship (see Figure 5.16 (c)), the sequence of input waveforms was scaled to have an amplitude of $\pm$5V with an offset of 5V, such that the input waveforms have values between 0 and 10 V. Equivalently, for the NARMA10 task, the sequence of random input values $u(k) \in [0, 0.5]$ was scaled by a factor of 10 and offset by 5V so that the input voltages have values between 0 and 10 V after masking.

The SRNDN was used to perform each task for many different cases (e.g. with/without delayed feedback etc) so to aid readability, the detailed results are presented in Appendix B and a summary of the main findings is given here.

Appendix B.1 presents the results for the waveform discrimination task: Appendix B.1.1 contains the results for the case where delayed feedback is included, while Appendix B.1.2 contains the results without delayed feedback. It was found that the SRNDN performs poorly at the waveform discrimination task, achieving a maximum classification score of only 51.5% with delayed feedback and 80% without delayed feedback. Unlike the MGO or the TRNDN which perform best with delayed feedback, task performance using the SRNDN *improved* when the delayed feedback mechanism was removed.

Appendix B.2 presents the results for the NARMA10 task. Appendix B.2.1 shows the delayed feedback case. It was found that the virtual reservoir could produce a good approximation of the training data (NRMSE, 0.084), but performed poorly for the testing data (NRMSE, 0.65). This is a characteristic sign of overfitting.

Appendix B.2.2 therefore summarises attempts to reduce overfitting by using ridge regression (rather than linear regression) to train the weights (see Section 4.3.5). It was found that the overfitting could be reduced, narrowing the difference between the training and testing performance, but that the testing performance remained poor with the lowest NRMSE value obtained being 0.453.

Appendix B.2.3 contains the results of the NARMA10 task without delayed feedback. The results are substantially the same as the case with delayed feedback in Appendix B.2.1 which suggests that the recurrence provided by the delayed feedback is not being utilized by the SRNDN. Overfitting was again observed and so training was also attempted using ridge regression. The outcome was essentially the same as the delayed feedback case: the difference between training and testing performance was reduced, but the testing performance remained poor.

Finally, Appendix B.3 considers two reasons for the generally poor performance of the SRNDN. The first reason is that, like the experimental switching rate data, the synthetic switching rate generated by the SRNDN features many empty bins ($x = 0$). The second reason considered is stochasticity. The response of the switching rate to a particular input voltage can vary over several orders of magnitude from trial to trial i.e. the switching rate is highly stochastic. In real PASNs, this stochasticity is produced by critical avalanche dynamics, and the SRNDN was designed to emulate such behaviour. Because SRNDN switching rate is used as the explanatory variable, it was hypothesised that the presence of zeros and stochasticity may be responsible for the poor performance of the SRNDN as compared to the MGO or TRNDN.

The effect of zeros in the explanatory variable x is investigated in Appendix B.3.1. The TRNDN was used to repeatedly perform the waveform discrimination task with an increasing proportion of x-values replaced with zeros. Because the TRNDN performs the waveform discrimination task very well (Appendix A.1.1), it is easy to see the significant degradation in performance as more and more x-values are replaced with zeros. It was found that when x contained many zeros, the case with delayed feedback performed much worse than the case without delayed feedback. This observation is consistent with the fact that the SRNDN performed worse at waveform discrimination when delayed feedback was included, the opposite trend observed for other NDN models. One explanation is that the delayed feedback mechanism provides recurrent connections which allow previous virtual node states to

remain in the reservoir. If some of these virtual node states are zeros, which do not reflect the input signal, then their prolonged retention by the reservoir leads to increased error.

The effect of stochasticity in the explanatory variable x was investigated in Appendix B.3.2 using a similar procedure. Instead of replacing values of x with zeros, all values of x were scaled by a random value r. Each r was drawn from a normal distribution centred at one. By repeating the waveform discrimination task using wider and wider normal distributions (i.e. increasing its standard deviation σ), the explanatory variable became increasingly stochastic. Consequently, the task performance deteriorated with increasing stochasticity, and the case with delayed feedback was more severely affected by the perturbations to x. Again, it is the presence of recurrent connections which allow perturbed virtual node states to influence the virtual reservoir state for longer, leading to worse performance for the case with delayed feedback.

5.3.3 Summary of results from the SRNDN

In summary, the SRNDN model did not perform well at either the waveform discrimination task or the NARMA10 task. It was also found that the delayed feedback mechanism had a negative effect on task performance. Both of these observations could be explained by the large number of zeros and the inherent stochasticity of the SRNDN explanatory variable. Section 5.3.1 demonstrates that the SRNDN synthetic switching rate is a good approximation of the behaviour of real PASNs in the switching regime, and so similarly poor performance is expected from a physical implementation of TDRC which uses the PASN switching rate as the explanatory variable.

5.4 Conclusions

Chapter 5 explored time-delay reservoir computing using several numerical models of nonlinear dynamical nodes: the Mackey-Glass oscillator, the tunnelling regime nonlinear dynamical node and the switching regime nonlinear dynamical node. In each case the waveform discrimination and NARMA10 tasks were performed, the number of virtual nodes N and response time T were optimised, and the effect of including/excluding the delayed feedback mechanism was investigated. The results are summarised in Table 6.

The MGO was the best performing numerical NDN at both the waveform discrimination and NARMA10 tasks with NRMSE values of 3.70×10^{-7} and 0.053 respectively. These rose to 0.247 and 0.157 respectively when the delayed feedback mechanism was removed, highlighting the significant role being played by direct recurrence.

The MGO also showed that in the absence of the direct recurrence provided by the delayed feedback mechanism, a long time constant could be used to facilitate indirect recurrence. For cases without the delayed feedback, the optimal result was found for relaxations times $T \approx \tau$ for the waveform discrimination task, and $T \approx 10\tau$ for the NARMA10 task. These optimum response times relate to the requirements of each task: waveform discrimination requires memory of at least one time step so that the NDN output depends on the *sequence* of input values, and NARMA10 at least ten time steps so that the next value in the sequence can be predicted. The fact that a long time constant can be used to provide recurrence may be helpful in cases where the implementation of a direct external feedback loop is challenging.

NDN	Waveform Discrimination (NRMSE)	Waveform Discrimination Score (%)	NARMA10 (NRMSE)
MGO with feedback	3.70×10^{-7}	100	0.053
MGO without feedback	0.247	100	0.157
TRNDN with feedback	6.97×10^{-7}	100	0.176
TRNDN without feedback	0.202	100	0.157
SRNDN with feedback	0.500	51.5	0.207
SRNDN without feedback	0.500	80	0.207
SRNDN with feedback (Ridge)	0.499	52.5	0.248
SRNDN without feedback (Ridge)	0.499	56.25	0.248

Table 6: Summary of numerical NDN results. NRMSE values are from the testing data. Shaded cells contain results from training via ridge regression. All other results were obtained by training with linear regression.

The TRNDN performed almost as well as the MGO for the waveform discrimination task with a NRMSE of 6.97×10^{-7}, however the TRNDN performed significantly worse at the NARMA10 task with a NRMSE of 0.176. When the delayed feedback was removed, the optimal performance decreased significantly for waveform discrimination, but only marginally for the NARMA10 task. This suggests that the TRNDN model may not be optimised for time series prediction tasks. Further optimisation of the parameter κ may improve the NARMA10 results.

Although the SRNDN produces a very convincing replication of the experimental switching rate, including the presence of avalanches and criticality, the SRNDN performed poorly at both tasks. The optimal results correspond to trivial cases where very large response times supress the switching activity to a minimum. This occurs because the stochastic spikes of activity in the switching rate do not map to particular input voltages (or sequences of them) and so the lowest errors correspond to cases where the switching rate is low.

How Should a Real PASN be Implemented as a Physical NDN?

Based on the results of the numerical TDRC simulations explored in this chapter, the optimal parameter choices for implementing a PASN as a physical NDN can be inferred.

The first choice is the operating regime; the TRNDN performed much better than the SRNDN because of the stochasticity of the switching rate, indicating that that a PASN should be operated in the tunnelling regime for best results. This requires using sub-threshold voltages as the inputs and measuring the conductance as a function of time. The time constant may be applied using a passive low-pass filter in the circuit, or it may be applied numerically in a post-processing step.

A delayed feedback mechanism should be used if possible, as this enabled excellent performance of the TRNDN at the waveform discrimination task. If a delayed feedback mechanism cannot be implemented, then a long time constant ($T \approx N\theta$ for waveform discrimination and $T \approx 10N\theta$ for NARMA10) should produce the best results.

Chapter 6

Experimental Implementation of TDRC

The two distinct regimes of PASN operation, tunnelling and switching, were modelled as numerical NDNs in Chapter 5. Results of both the waveform discrimination and NARMA10 tasks were used to assess the TDRC parameter choices for each NDN model and each task. The TRNDN model indicated that a real PASN in the tunnelling regime may perform well at classification and time series prediction tasks. Conversely, the SRNDN showed that PASNs in the switching regime are unlikely to perform well at either task because of the inherently stochastic switching rate. Hence, this chapter presents experimental results from attempts at both the waveform discrimination and NARMA10 tasks using a real PASN in the tunnelling regime. In this thesis, TDRC was not performed using real PASNs in the switching regime.

6.1 Tunnelling Regime

Section 1.4 describes the tunnelling and switching regimes of PASN response to applied voltages V. Recall that the tunnelling regime is accessed at voltages below the switching threshold voltage V_{thres}, and that V_{thres} can be increased by applying $V \gg V_{thres}$ for extended periods of time. The use of a real PASN in the tunnelling regime as a NDN for TDRC will hereafter be referred to as a *physical* TRNDN to distinguish from the numerical TRNDN model presented in Section 5.2.

Input Signals

When using the physical TRNDN, input signals are encoded as applied voltages, and so the threshold voltage V_{thres} sets an upper limit to the range of inputs which may be applied without inducing unwanted switching events. Hence, to obtain maximum input range and to fully exploit the nonlinear G-V characteristic, PASNs were fatigued by applying 10 V DC for several days before being used as physical TRNDNs (see Section 1.4.6. This process caused the switching threshold voltage to increase beyond 8 V, and so input signals could be applied within the range 0-8 V without inducing unwanted switching events.

Input signals $J(t)$ were applied as voltages across the two electrodes of the PASNs using an arbitrary waveform generator. The input signals for the waveform discrimination task were sine and square waveforms of 1 second period, each with 8 time steps τ, and each timestep with $N = 200$ mask elements[17] of duration θ =625 μs.[18] The sine and square waveforms had an amplitude of $\pm$3.5 V and an offset of 4.5 V so that the input signals ranged between 1-8 V. For the NARMA10 task, each random input value $u(k)$ was applied for one timestep ($\tau = 100$ ms) each containing $N = 200$ mask elements of $\theta = 500$ μs. The amplitude of $u(k)$ was scaled by a factor of 7 and offset by 4.5 V so that again, the input signals range over 1-8 V.

[17] $N = 200$ was chosen for the experimental implementation because the results from the numerical TRNDN model showed good performance at waveform discrimination for $N = 200$.

[18] When using a physical NDN which has a well-defined response time T, the interval separating virtual nodes θ must be carefully chosen. However, the natural response time of the PASN is not used in this thesis and an effective response time (low-pass filter) is applied in post-processing. This procedure and the reasons for it are discussed in detail below. Hence, $\theta = 625\mu$s (and 500μs for NARMA10) was chosen for convenience.

Explanatory Variable

The explanatory variable which defines the state of the physical TRNDN is the conductance G. G is a nonlinear function of applied voltage V, as shown in Figure 5.13. G is obtained by conducting time-resolved measurements of V and the resulting current I and then calculating $G = I/V$. Conductance measurements (Section 2.2.2) were made at intervals of 10 μs and averaged over 2 points to reduce noise, resulting in a sample of G every 20 μs as shown by the blue curve in Figure 6.1. Further noise reduction was achieved by passing G through a 25 point median filter, represented by the red curve in Figure 6.1. The states of the virtual nodes were then found by sampling G at intervals of θ such that the explanatory variable $\tilde{x} = G(t = n\theta)$ where $n = 1 \ldots NK$, N is the number of virtual nodes in the delay line and K is the total

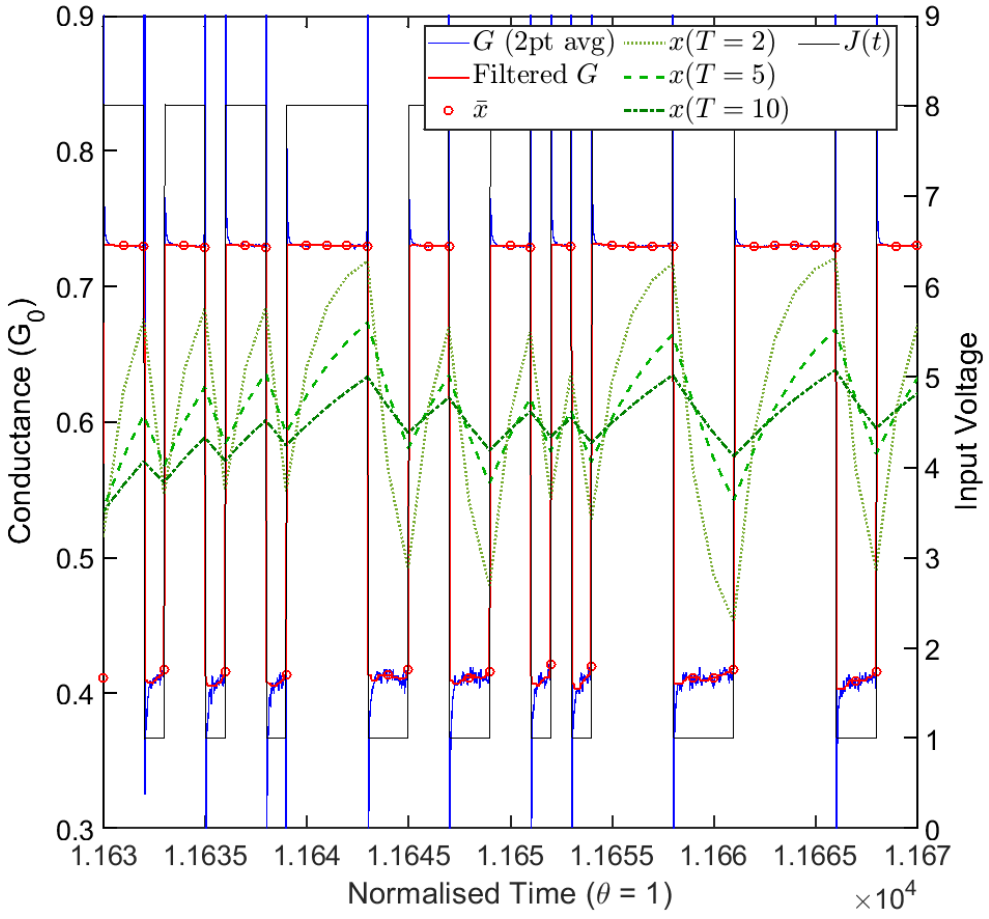


Figure 6.1: Physical TRNDN sampling procedure. A measurement of the conductance G of a PASN (blue) in response to the applied voltage (black) after 2-point averaging to reduce noise. Noise levels were further reduced using a median filter (red curve) before sampling at intervals of θ (time is normalized so that $\theta = 1$) denoted by the red markers. The three different green curves represent the filtered and down-sampled G after having different response time constants applied (see text). Small time constants produce faster variations in G and larger oscillation amplitude while large time constants lead to slower responses and a damped oscillation amplitude.

number of time steps (τ) in the input sequence. The $\sim$ above x denotes the fact that the time constant T is yet to be applied (see below) and that the values in $\tilde{x}$ are not the final virtual node states. $\tilde{x}$ is represented by the red markers in Figure 6.1.

Response Time

There is a natural response time of the PASN, which is on the order of ~100 μs. This response time could be used to provide the connectivity between virtual nodes if the measurements of V and I are sampled at a sufficiently high bandwidth. Unfortunately, the measurement circuit used in this study produced transient current spikes in response to voltage step edges; likely due to a combination of parasitic capacitance and impedance mismatches in the circuit.

This problem is illustrated in Figure 6.2 which shows a measurement of G (blue) and the applied voltage V (black). The step edges in V cause artefacts in the measurement of I (and consequently G) which appear as large spikes. If the spikes were not present, then the response time would allow a gradual transition from G_1 to G_2 along a path approximated by the dashed line in Figure 6.2. This would mean that if $\theta \lesssim T$, the virtual node states would depend on each other, as per the discussion in Section 4.3.4.1 and the example in Figure 4.7. $\tilde{x}$ would then be equal to x because no time constant would need to be applied in post-processing. However, the spike in G at the voltage step edge means that the conductance is discontinuous, and the response time causes G to follow the path highlighted by the solid red curve in Figure 6.2. Consequently, the natural response time of the circuit could not be used to provide correlation between virtual node states.

The response time was therefore applied in a post-processing step using

$$x_i = x_{i-1} + \frac{1}{T}(\tilde{x}_i - x_{i-1}) \tag{64}$$

where T is the response time constant and x is the explanatory variable with the time constant applied. x is represented by the green curves in Figure 6.1 for three different values of T ($2\theta, 5\theta, 10\theta$). While each value of $\tilde{x}$ depends only on the input voltage and is independent of previous values of $\tilde{x}$, the time constant has the effect of making each value in x depend on the preceding values of x as in the case of the numerical TRNDN model. This operation is essentially a low pass-filter which is commonly used in TDRC implementations [235].

It is desirable to use the natural response time of the PASN as T in order to utilize more of the PASN functionality and to reduce the amount of post processing (and power consumption). However, applying the time constant in post-processing has the significant advantage of being able to explore T as a free parameter by applying many different time constants to the same measurement data and determining which gives the best results. To perform the same analysis using the natural response time as T would require repeating the measurements using many different values of θ. Hence there is an opportunity to efficiently compare task performance as a function of the time constant with that of the numerical TRNDN.

Partial Delayed Feedback Mechanism

To use a delayed feedback mechanism with a physical TRNDN requires that signals are analysed in real time and reinjected into the PASN in the next time step. This means that the noise reduction, down-sampling and application of the time constant must be performed

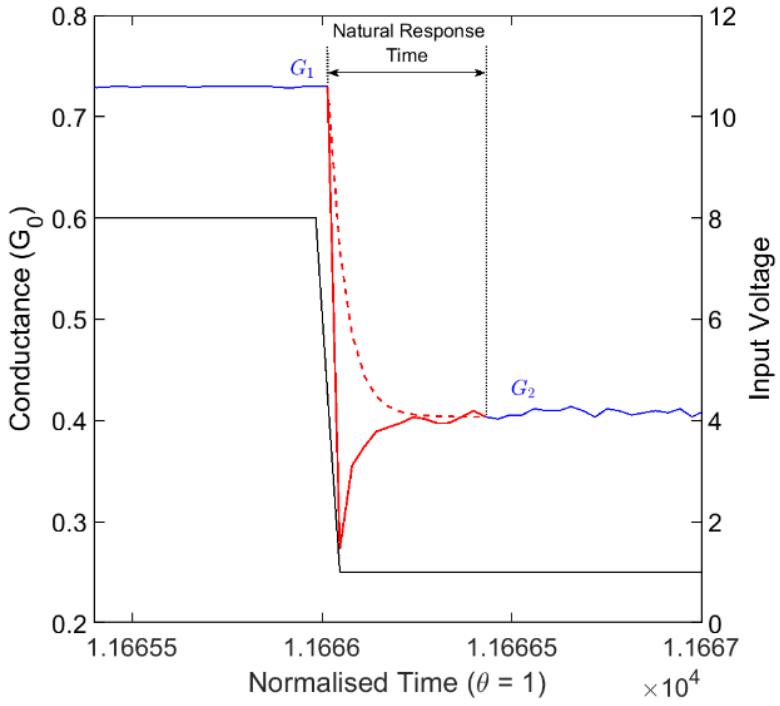


Figure 6.2: Physical TRNDN transient current spikes. A measurement of the conductance G of a PASN (blue/red) showing a transient spike in response to a step in the applied voltage (black). The presence of the spike and the natural response time of the PASN circuit causes the conductance to follow the path highlighted by the solid red curve. This path is undesirable because there is a discontinuity between G_1 and G_2. If the spike were absent, then the conductance would follow a path like the one represented by the dashed line and θ could be tuned so that virtual node states were correlated.

during each time step so that the node states can be used to augment the input voltage signal which is applied in the next time step. Such in-line data processing and waveform generation is possible; however it would require the development of a specialized instrument control program which is outside the scope of this thesis (see Section 6.3 on future work).

Hence, a "partial delayed feedback" mechanism was developed. Partial delayed feedback is performed in post-processing and involves combining virtual node states x_i with the state of the same virtual node from the previous time step x_{i-N}, using

$$x_i \rightarrow \frac{x_{i-N} + x_i}{2} \tag{65}$$

Partial delayed feedback has a similar effect to the full delayed feedback used in the numerical simulations; it makes the state of each virtual node depend on its previous state by passing the previous state directly across time steps i.e. recurrent connectivity. The key difference is that the previous state of the virtual node is not reinjected into the NDN, and so is not subject to the nonlinear transformation. Consequently, the partial delayed feedback mechanism does not result in the same level of recurrence as the full delayed feedback mechanism. The partial delayed feedback mechanism was tested using the MGO numerical implementation of TDRC and was found (results not shown) to provide a significant improvement in performance (over simulations with no delayed feedback) but did not perform as well as the full delayed feedback.

6.1.1 Waveform Discrimination using a Physical TRNDN

This section presents the results of the waveform discrimination task which was performed using a physical TRNDN with $N = 200$ virtual nodes. A wide range of different time constants were used to find the optimum T and cases with and without partial delayed feedback were investigated.

Example Result with Partial Delayed Feedback

An example result of the waveform discrimination task is shown in Figure 6.3 for $N = 200$ and $T = 5$. A subsection of the input sequence $J(t)$ is shown in Figure 6.3 (a). In the case of a *physical* TRNDN, $J(t)$ is the voltage applied across the two electrodes of the PASN and

can be measured directly as in Figure 6.3 (a). Figure 6.3 (b) shows the explanatory variable x of the physical TRNDN which is the measured conductance response of the PASN after postprocessing using $T = 5$ (see Figure 6.1). The effect of partial delayed feedback can be seen in the difference between the physical TRNDN response on the left and right side of Figure 6.3 (b) where the input signals are the same. Figure 6.3 (c) shows the virtual reservoir output $\hat{y}$ (blue) and the average of $\hat{y}$ over each waveform cycle $\langle \hat{y} \rangle$ (black) superimposed with the target function y (red) for the full sequence of 600 sine/square waveforms. A zoomed region of Figure 6.3 (c) is shown in Figure 6.3 (d), where $\hat{y}$ can be seen fluctuating about y.

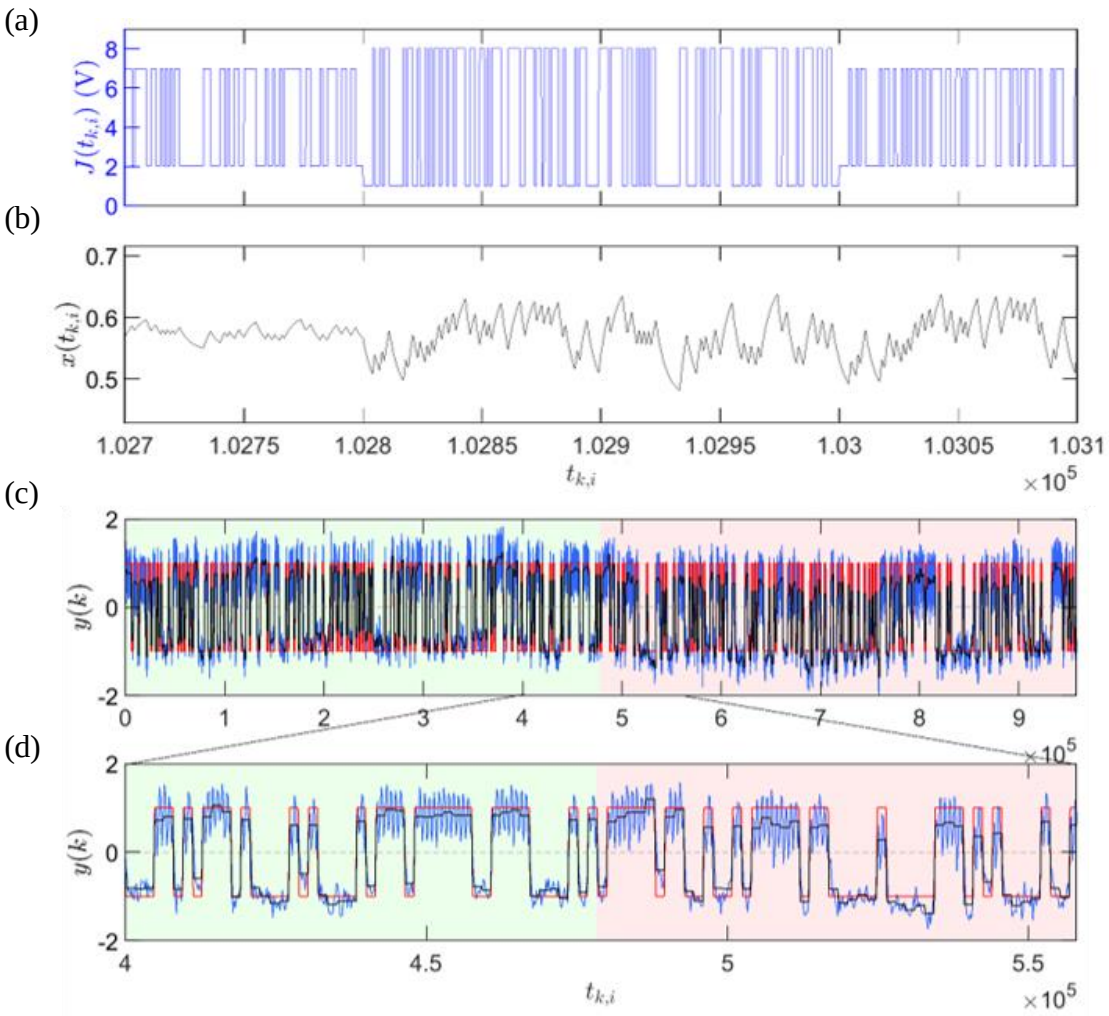


Figure 6.3: Waveform discrimination task using physical TRNDN with partial delayed feedback; $N = 200, T = 5\theta$. (a) A subsection of the input sequence $J(t)$. (b) The tunnelling regime response of the PASN to the stimulus shown in (a) using a response time $T = 5\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.215 and 0.259 for the training and testing data respectively. (d) A zoomed region of (c) showing clearly that the two waveform classes have been successfully separated. The classification is made by comparing the average of $\hat{y}$ over each waveform (black) with the decision boundary at zero.

There is a clear separation between the values of $\hat{y}$ corresponding to the different waveform classes. All 600 waveforms were correctly classified giving a classification score of 100% and NRMSE values of 0.215 and 0.259 for the training and testing data respectively.

The Effect of Partial Delayed Feedback

The waveform discrimination task was performed both with and without the partial delayed feedback mechanism described by Eq. (65). Recall that delayed feedback aims to provide recurrence to the virtual reservoir. Recurrence causes the virtual reservoir to respond differently to different *sequences* of input signals $u(k)$, rather than producing a response which is identical for the same input value u regardless of the sequence in which it is embedded.

In the same way that Figure 5.7 illustrates the effect of recurrence from the inclusion of the delayed feedback mechanism with the MGO, Figure 6.4 demonstrates the effect of the recurrence induced by the inclusion of the partial delayed feedback mechanism with the physical TRNDN. Figure 6.4 (a) contains a sequence of input amplitude values in $I(t)$

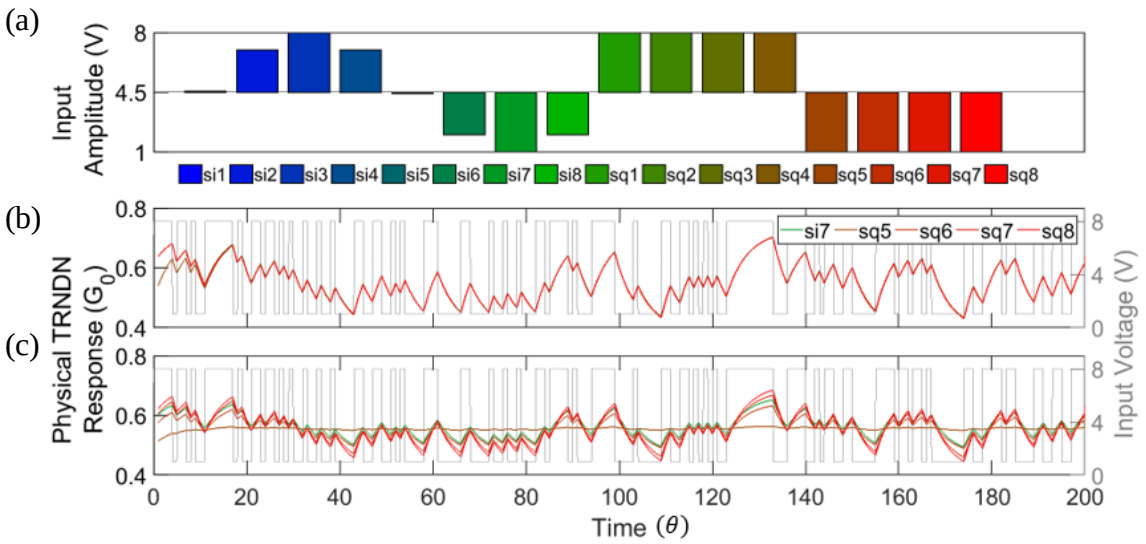


Figure 6.4: The effect of recurrence from partial delayed feedback. (a) A sequence of amplitude values in $I(t)$ corresponding to a sine wave followed by a square wave. Each amplitude value is applied for one time step τ, by multiplying by the mask function $M(t)$ which is shown in grey in (b, c). (b) The response of the physical TRNDN *without* partial delayed feedback ($T = 5\theta$) over different time steps which have an input amplitude of minus one. The response curve for time step si7 is not resolvable from the sq6-8. (c) The response of the physical TRNDN *with* partial delayed feedback ($T = 5\theta$) over the same time steps shown in (b). The response curve for si7 is now resolvable from the sq5-8 curves along the entire delay line. This illustrates the increased separability of different classes when the virtual reservoir has recurrence provided by the partial delayed feedback mechanism.

corresponding to a sine wave followed by a square wave. There is one time step in the sine wave and four time steps in the square wave which have an amplitude of 1 V. If the virtual reservoir has no recurrence, it does not retain information from previous input values, and will hence respond in the same manner to all five inputs of 1 V, regardless of their place in the sequence. The physical TRNDN response to all five of the input values of 1 V is shown in Figure 6.4 (b) without partial delayed feedback and in Figure 6.4 (c) with partial delayed feedback. The delay line has a duration $\tau = N\theta$ where $N = 200$, which is imposed by the mask function $M(t)$ (grey line in each panel). In both cases, the physical TRNDN has a time constant of $T = 5\theta$. In Figure 6.4 (b) where there is no partial delayed feedback, the five response curves, each corresponding to a different time step in the sequence shown in Figure 5.7 (a), are almost identical. In contrast, the five response curves in Figure 6.4 (c) are distinguishable along the entire delay line because of the recurrence provided by the partial delayed feedback mechanism. More important for the classification task, the si7 curve is separated from all of the sq curves indicating that the partial delayed feedback mechanism provides the virtual reservoir with recurrence, even though past virtual node states are not reinjected into the NDN.

Optimal Response Time for Waveform Discrimination with Partial Delayed Feedback

The waveform discrimination task was repeated using the physical TRNDN with $N = 200$ and a range of different response times applied in postprocessing. Figure 6.5 shows the performance (NRMSE, left axis; classification score, right axis) as a function of the applied time constant T for (a) the case with partial delayed feedback, and (b) the case without partial delayed feedback.

The case *with* partial delayed feedback shows a minimum NRMSE of 0.248 for $T = 130\theta = 0.65\tau$. This optimum result is presented in Appendix C.1. In contrast, the numerical TRNDN for $N = 200$ with delayed feedback resulted in an optimal NRMSE of 6.97×10^{-7} for an optimal time constant $T = 4\theta = 0.02\tau$ (Figure A.2).

One reason for the difference between the experimental and numerical results is the presence of noise in the measurement of the PASN conductance which is not present in the numerical model. Measurement noise has a similar effect to the stochasticity explored in Appendix B.3.2 because it causes the explanatory variable x to contain stochastic variations.

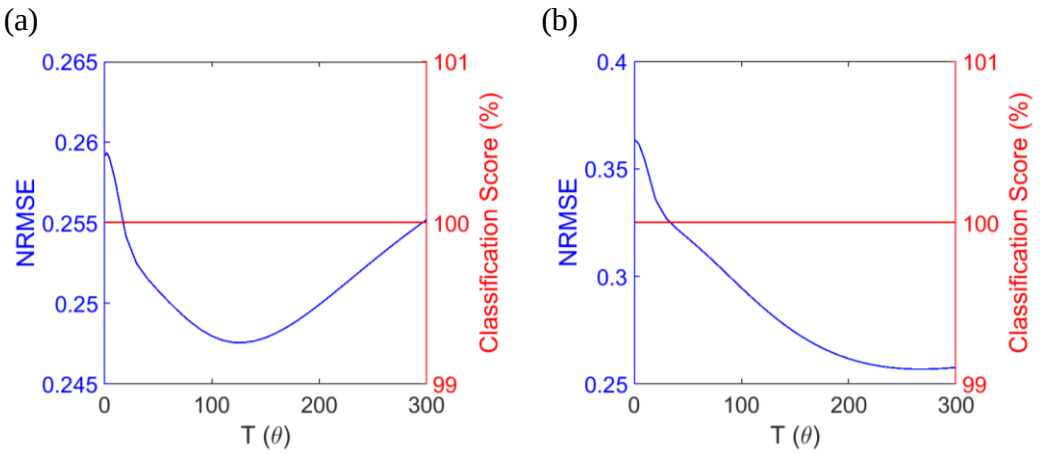


Figure 6.5: Optimising the time constant for the waveform discrimination task using the physical TRNDN. The NRMSE (blue) and the classification score (red) as a function of the response time T for the physical TRNDN. (a) shows the case with partial delayed feedback and (b) shows the case without partial delayed feedback. Best results are achieved when partial delayed feedback is included, though the difference between the lowest NRMSE in each case is small (~0.01) and the classification score is 100% for all T both with and without feedback.

It was shown in Figure B.14 that when the delayed feedback mechanism is included, stochasticity in x leads to a significant degradation in task performance both in terms of NRMSE and classification score. Thus, it is likely that the measurement noise in the explanatory variable of the physical TRNDN is responsible for the larger NRMSE than its numerical counterpart.

Another reason that the physical TRNDN produces a larger NRMSE than the numerical TRNDN model is the difference between the recurrence provided by the full and partial delayed feedback mechanisms. The partial delayed feedback mechanism provides a weaker recurrence than the full delayed feedback mechanism. This is because the previous virtual node states are not reinjected into the NDN, but rather combined with the present output using Eq. (65). Consequently, the information passed through the recurrent connection in the virtual reservoir is not subject to a nonlinear transformation as in the case of the full delayed feedback mechanism. NDNs with partial delayed feedback therefore perform worse than those with full delayed feedback, but still better than those without delayed feedback (which are essentially feed-forward virtual reservoirs).

Finally, the longer optimal response time for the physical TRNDN ($T = 0.65\tau$) as compared with the numerical TRNDN ($T = 0.02\tau$) can be attributed to the weaker recurrence of the partial delayed feedback mechanism. In Section 5.1, it was shown that optimal time constants were found to be much longer in the absence of the direct recurrence provided by the full delayed feedback because indirect recurrence could be facilitated by response times

which spanned time steps. The optimal time constant is therefore longer for the physical TRNDN because it makes up for the weaker recurrence of the partial delayed feedback mechanism.

Optimal Response Time for Waveform Discrimination without Partial Delayed Feedback

Figure 6.5 (b) shows that the physical TRNDN *without* partial delayed feedback produced a minimum NRMSE of 0.257 at $T = 270\theta = 1.4\tau$. This optimal result is presented in Appendix C.1. The optimal result from the physical TRNDN without delayed feedback can be compared with that of the numerical TRNDN (Figure A.4 (a)), where optimal parameters $N = 40$ and $T = 70\theta = 1.75\tau$ produced a minimum NRMSE of 0.202.

The optimal response times are both consistent with the dark band in Figure A.4 (a) which represents a region of the parameter space which results in good task performance. The location of the dark band corresponds to response times which are about the same length as the delay line because this allows some information to be passed between time steps without the direct recurrent connections provided by the delayed feedback mechanism (discussed further in Section 5.1).

Again, the physical TRNDN produces an optimal NRMSE (0.257) which is slightly larger than that of the numerical TRNDN (0.202). This difference can be attributed to the presence of noise in the explanatory variable x which arises from the measurement of the PASN conductance.

6.1.2 NARMA10 using a Physical TRNDN

This section presents the results of the NARMA10 task which was performed using a physical TRNDN with $N = 200$ virtual nodes. A wide range of different time constants were used to find the optimum T and cases with and without partial delayed feedback were investigated.

Example Result with Partial Delayed Feedback

An example result of the NARMA10 task is shown in Figure 6.6 for $N = 200$ and $T = 5\theta$. Figure 6.6 (a) shows a subsection of the measured input sequence $J(t)$ containing ten of the random input values $u(k)$. Each input value is masked and applied as a sequence of voltages

across the two PASN electrodes. Figure 6.6 (b) shows the conductance response of the PASN x to the input in Figure 6.6 (a) after having been postprocessed in accordance with the procedure outlined in Figure 6.1 using $T = 5\theta$. Figure 6.6 (c) shows the virtual reservoir output $\hat{y}$ (blue) overlaid with the target function y (red) for the full sequence of 1000 input values $u(k)$. A zoomed region of Figure 6.6 (c) is shown in Figure 6.6 (d), where $\hat{y}$ can be seen following y closely for the training data (left) but less closely for the testing data (right). The respective NRMSE values which quantify the task performance are 0.084 and 0.265, reflecting the much better approximation of y during training than in testing.

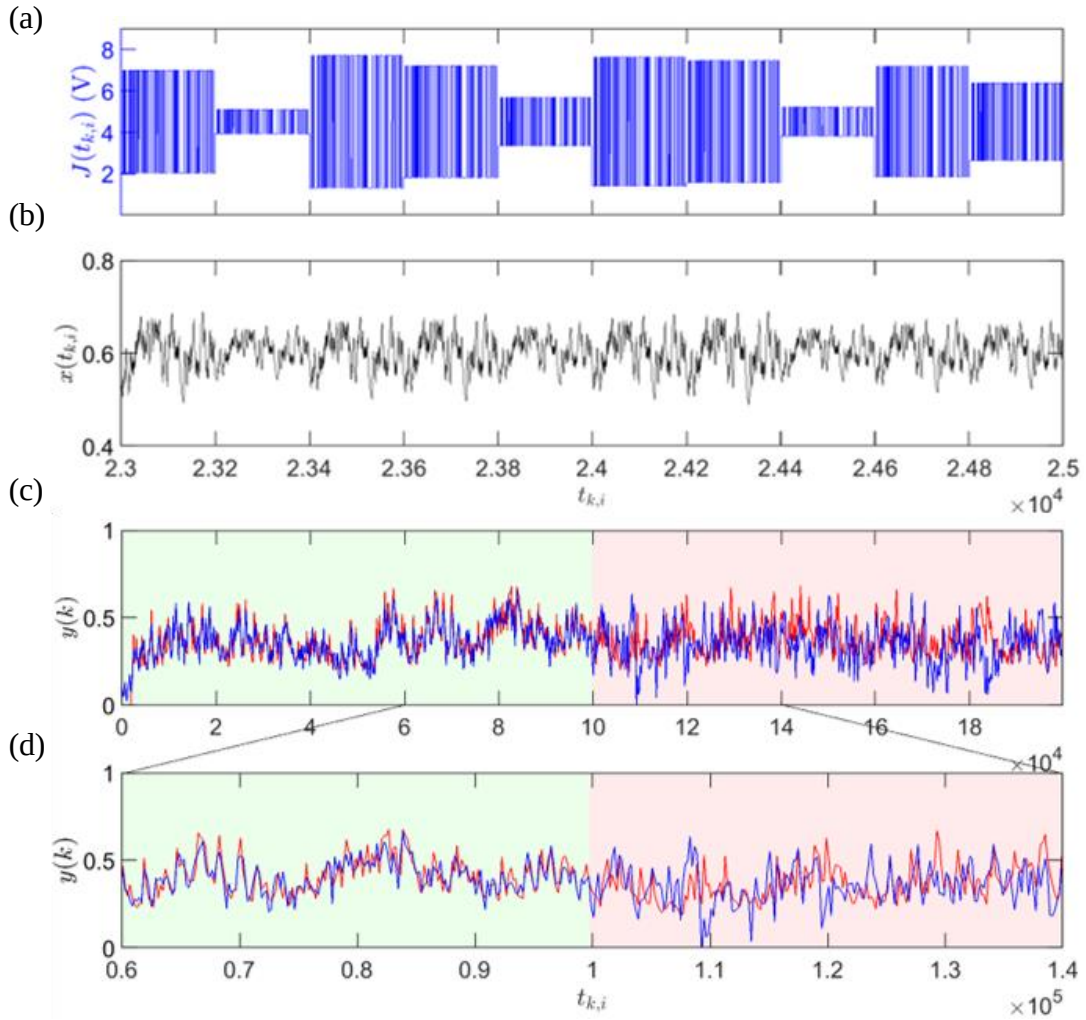


Figure 6.6: NARMA10 task using physical TRNDN with partial delayed feedback; $N = 200, T = 5\theta$. (a) A subsection of the input sequence $J(t)$. (b) The tunnelling regime response of the PASN to the stimulus shown in (a) using a response time $T = 5\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.084 and 0.265 for the training and testing data respectively. (d) A zoomed region of (c) showing $\hat{y}$ and y in more detail. $\hat{y}$ is a good approximation of y for the training data, but less so for the testing data: a sign of overfitting.

The significant difference between the training and testing NRMSE values signifies overfitting. As in the case of the numerical SRNDN in Appendix B.2, ridge regression can be used to train the weights in order to reduce overfitting. The weights were re-trained using ridge regression, the results of which are presented in Appendix C.2.1. It is clear that while there is a reduction in the difference between the training and testing NRMSE values, the solution when training via ridge regression is trivial: the weights have been reduced to very small values so that $\hat{y}$ becomes a flat horizontal line which passes through the mean of the target function y. The variations in $\hat{y}$ are so small that it does not approximate the chaotic variations of the NARMA10 time-series, and training via ridge regression is deemed unsuccessful.

Optimal Response Times

The NARMA10 task was repeated using the physical TRNDN with $N = 200$ and a range of different response times applied in postprocessing. Figure 6.7 shows the NRMSE as a function of the applied time constant T for (a) the case with partial delayed feedback, and (b) the case without partial delayed feedback.

The case *with* partial delayed feedback shows a minimum testing NRMSE of 0.264 at $T = 2\theta = 0.01\tau$. This optimum result is shown in detail in Figure C.4 in Appendix C. The case *without* partial delayed feedback shows a minimum testing NRMSE of 0.253 at $T = 140\theta = 0.7\tau$. This optimum result is shown in detail in Figure C.5 in Appendix C.

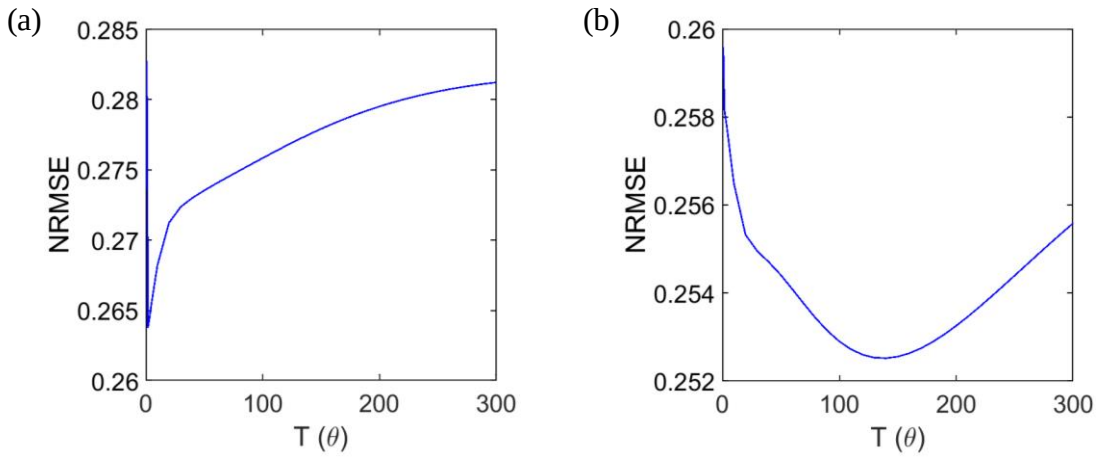


Figure 6.7: Optimising the time constant for the NARMA10 task using the physical TRNDN. The NRMSE as a function of the response time T for the physical TRNDN. (a) shows the case with partial delayed feedback and (b) shows the case without partial delayed feedback. Best results are achieved when partial delayed feedback is excluded, though again the difference between the lowest NRMSE in each case is small ($\sim$0.01).

The performance of the *numerical* TRNDN at the NARMA10 task (Figure A.7 (a) and Figure A.9 (a)) showed a moderately weak dependence on the response time with NRMSE varying by ~0.04. A slightly weaker dependence is observed here for the *physical* TRNDN, with NRMSE values varying by < 0.02. The typical trend of a longer optimal response time in the absence of delayed feedback is again observed, due to the recurrence which can be facilitated by long time constants. The slightly larger NRMSE for the physical TRNDN arises again due to noise in the measurement of the PASN conductance which serves as the explanatory variable x.

6.1.3 Comparison with Literature Results

In the previous sections, the physical TRNDN with $N = 200$ was used to perform the waveform discrimination and NARMA10 tasks. Each task was performed with and without the partial delayed feedback mechanism described in Section 6.1, and using a range of response time constants. The best result for each case and the corresponding response times are summarised in Table 7. For both tasks, the inclusion/exclusion of the partial delayed feedback mechanism does influence the optimal response time but has no significant effect on the NRMSE values. This indicates that the partial delayed feedback mechanism is an ineffective method of providing recurrence to the virtual reservoir of the physical TRNDN.

Table 1 contains the results of the waveform discrimination task from the reviewed literature in Section 4.4. As the physical TRNDN produces classification scores of 100%, it performs better than all of the literature results except one: Ref. [201] which reports 100% classification score as well as a test NRMSE of 0.0387. This is a very low error considering that it is an optoelectronic implementation which is subject to real-world effects such as noise. It should also be noted that Ref. [201] used a smaller virtual reservoir of only $N = 50$ However, Figure A.4 (a) shows that the numerical TRNDN did produce an even smaller

Partial Delayed Feedback?	Waveform Discrimination			NARMA10	
	NRMSE	Score (%)	T	NRMSE	T
Yes	0.248	100	130 θ	0.264	2 θ
No	0.257	100	270 θ	0.253	140 θ

Table 7: Summary of the optimised experimental TDRC results using a physical TRNDN with $N = 200$.

NRMSE of 6.97×10^{-7} for $N = 200$ when the full delayed feedback mechanism was included, and that similarly good performance could be achieved over a wide range of N. This indicates that the physical TRNDN may be able to surpass the performance of Ref. [201] if the full delayed feedback mechanism is implemented experimentally, and if measures are taken to minimize sampling noise (see Section 6.3 for future work).

Table 5 contains the results of the NARMA10 task from the reviewed literature in Section 4.4. For $N = 200$, the results obtained using the physical TRNDN are surpassed by those of Refs. [205,238]. However, both of these results are from numerical implementations and are comparable to the results obtained using the numerical TRNDN (see Table 6). The NARMA10 task requires strong recurrence in order to retain enough information from ten previous time steps so that the next value may be accurately predicted. Unfortunately, the partial delayed feedback mechanism was unable to provide the required recurrence to obtain good performance. Again, this suggests that if the full delayed feedback mechanism can be implemented experimentally and sampling noise can be reduced, then the physical TRNDN may be capable of surpassing the results reported in the literature.

6.2 Conclusions

Chapter 6 presented the results of an experimental implementation of TDRC using a PASN in the tunnelling regime as the NDN i.e. a physical TRNDN. The waveform discrimination and NARMA10 tasks were used to benchmark the performance of the physical TRNDN. Due to experimental constraints, the full delayed feedback mechanism utilized in the numerical NDN implementations (Chapter 5) and in other works [201,202,204,210] could not be implemented, and so an alternative scheme called partial delayed feedback was implemented in postprocessing. The efficacy of the partial delayed feedback mechanism was investigated, and the response times were optimized for each case. The final results of the physical TRNDN task performance are presented in Table 7.

The physical TRNDN performed the waveform discrimination task well, surpassing the majority of reviewed literature in terms of NRMSE and providing 100% classification scores both with and without partial delayed feedback. Long response times ($T \sim \tau$) were found to be optimal because they facilitate recurrence within the virtual reservoir which could unfortunately not be facilitated by the partial delayed feedback mechanism.

The physical TRNDN did not perform well at the NARMA10 task with NRMSE values approximately 25% larger than those reported in the literature for the same sized virtual reservoir ($N = 200$). This is likely a consequence of the lack of recurrence provided by the partial delayed feedback mechanism and the presence of sampling noise in the measurement of the explanatory variable x.

The results presented in Chapter 6 mark the first ever neuromorphic implementation of a PASN and are a step towards more advanced architectures such as those utilizing multi-contact PASN devices.

6.3 Future Work

This Section presents plans for future work on utilizing PASNs for reservoir computing (RC) applications. Section 6.3.1 details improvements to the physical TRNDN architecture from Section 6.1, and Section 6.3.2 considers how a multi-contact PASN might be used to perform RC.

6.3.1 Improvements to the Physical TRNDN

Full Delayed Feedback Mechanism

The *partial* delayed feedback mechanism was developed as a work-around for the inability to implement a *full* delayed feedback mechanism experimentally. Partial delayed feedback is implemented in postprocessing following data collection. Hence the previous state of the virtual reservoir is not reinjected into the NDN and is not subject to a nonlinear transformation. Unfortunately, the results of Chapter 6 show that the partial delayed feedback mechanism provides insufficient recurrence to the virtual reservoir. This fact is evident by the similar NRMSE values obtained when partial delayed feedback is included/excluded (e.g. see Table 7). The partial delayed feedback mechanism does affect the dynamical response of the NDN, as apparent from the very different optimal response times, but the dependence of task performance on response time is weak. Consequently, very little performance enhancement is achieved by including partial delayed feedback.

It is therefore required that the full delayed feedback mechanism be implemented experimentally in order to facilitate direct recurrent connections in the virtual reservoir, and to obtain the corresponding performance enhancements. This is no small task, as it requires

in-line processing of measured data and subsequent re-injection of processed data into the PASN. Specifically, a program must be developed which can determine virtual node states from measured data in real time, and then augment the input signal $J(t)$ (i.e. the applied voltage) to be applied in the next time step. This might be achieved using laboratory control software developed in LabVIEW which could be naturally extended from existing software.

Noise Reduction

Appendix B.3.2 showed that the induction of random variations in the explanatory variable resulted in a significant degradation in task performance, especially if delayed feedback is included. This is because the NDN response to similar inputs becomes different from example to example (even if the nominal examples are identical e.g. sine waves), but the same set of weights is used to classify each example.

Sampling noise incurred in measuring the conductance of the PASN has exactly this effect and is thus responsible for the poorer performance of the physical TRNDN without delayed feedback as compared with its numerical counterpart. Thus, it is imperative that measures be taken to reduce sampling noise as much as possible. Furthermore, if the full delayed feedback mechanism is implemented experimentally as outlined above, these measures must be compatible with the in-line data processing requirements. This might be achieved by: measuring at a higher bandwidth and averaging over multiple points; using alternative measurement apparatus with advanced sampling functionalities; utilizing a four-terminal (4T) measurement set-up; using a passive low-pass filter in the measurement circuit. The last of these options is appealing because, not only does it remove noise filtering tasks from the programming environment (reducing power consumption and latency), but if carefully designed, the passive filter may also be used to tune the natural response time of the circuit (discussed below).

Utilizing the Natural Response Time

Figure 6.1 demonstrates the sampling procedure used to obtain the virtual node states from conductance measurements. This procedure involves applying the time constant T to changes in conductance, emulating a natural response time and thus providing correlation between the virtual node states. The reason that the time constant had to be applied in post processing rather than using the natural response time of the circuit is because of the discontinuity induced by artificial spikes in the conductance which coincide with step edges in the applied

voltage (see Figure 6.2). These transient current spikes are likely caused by parasitic capacitance/inductance within the measurement circuit, inducing a brief surge of current when the voltage slew rate is high (e.g. at step edges in the applied voltage). It is vital that this issue be mitigated in order to utilize the natural response time of the PASN circuit.

Once parasitic circuit elements have been removed from the measurement set-up, the transient response of the PASN conductance to stepwise changes in applied voltage should follow the dashed red line in Figure 6.2, rather than the solid red line featuring the spike. The response time of the circuit may then be tuned by the addition of a passive low-pass filter (e.g. an RC filter). Using the circuit response time of the circuit is beneficial because it removes the task of low-pass filtering from postprocessing allowing for the required in-line data processing and has the benefit of filtering out unwanted sampling noise. Furthermore, it utilizes more properties of the physical hardware, relying less on digital data processing, which is a primary goal of neuromorphic approaches.

6.3.2 Reservoir Computing with Multi-contact PASNs

TDRC was chosen for this study because it can utilize the dynamics of a single circuit element, which in this case is a two-contact PASN. Conventional RC on the other hand requires a multi-contact geometry in order to extract a higher dimensional representation of the input signal. While TDRC has been capable of demonstrating the first instance of neuromorphic applications using PASNs, it does not exploit the most novel features of PASNs e.g. the correlated and critical dynamics of the nanoparticle network which were demonstrated in Chapter 3.

To make use of the internal network dynamics, it is recommended that multi-contact PASNs be studied for RC applications. Conventional RC could be carried out by using one or more electrodes for injecting input signals and treating each of the remaining electrodes as a node within a reservoir. Node states could then be sampled *simultaneously*, rather than sequentially as in the case of TDRC, and the node states would be correlated due to the internal network dynamics. This architecture would be a much more impactful step towards neuromorphic computation because it would utilize unique properties of the percolating nanoparticle network, rather than the simple nonlinear dynamics which are shared by other less remarkable circuit elements e.g. diodes were used in Ref. [202].

Components of this study could also be included in a multi-contact PASN RC architecture. For example, it may be difficult to fabricate PASN devices which have a large number of electrodes N_{e}. In this case the number of nodes in the reservoir N would be limited by the available number of electrodes which may lead to sub-optimal task performance if N is too low. N may then be increased by combining the TDRC approach of multiplexing (i.e. applying a mask of length N_{mask}), with the conventional RC approach of simultaneously sampling multiple node states. Each electrode could be treated as a NDN and multiplexed to produce a virtual reservoir of virtual nodes. There would then be a virtual reservoir for each electrode, and the virtual reservoir states would be correlated by the internal dynamics of the percolating network. In this case, N could be increased from N_{e} to $N_{\mathrm{e}} \times N_{\mathrm{mask}}$, thus providing increased dimensionality of the higher-dimensional representation of the input signal.

Appendix A - Numerical TRNDN Results

This Appendix describes the performance of the tunnelling regime nonlinear dynamical node model (TRNDN) for the same waveform discrimination and NARMA10 tasks considered elsewhere in the thesis.

A.1 Waveform Discrimination

A.1.1 With Delayed Feedback

Example Result with Delayed Feedback

Figure A.1 presents an example result of the waveform discrimination task using the TRNDN *with* delayed feedback. Figure A.1 (a) shows a subsection of the input sequence $J(t)$ and Figure A.1 (b) shows the corresponding response of the TRNDN i.e. the explanatory variable x. Figure A.1 (c) contains the target function y (red) and virtual reservoir output $\hat{y}$ (blue) for the entire sequence of 400 waveforms. The TRNDN performs the task so well that the NRMSE between y and $\hat{y}$ is $\sim 5\times10^{-6}$. Consequently, y is completely obscured by $\hat{y}$ on this scale and so a zoomed region of Figure A.1 (c) is presented in Figure A.1 (d). The black line in Figure A.1 (d) is $\langle\hat{y}\rangle$: the average of $\hat{y}$ over each waveform cycle (8 time steps τ). $\langle\hat{y}\rangle$ is used to conduct the final classification by classifying waveforms with positive $\langle\hat{y}\rangle$ as sine and negative $\langle\hat{y}\rangle$ as square: a procedure called "squashing" the output. Output squashing is required because each waveform produces 8 values in $\hat{y}$ (corresponding to the 8 time steps in each waveform), which must be used to infer a single class: sine or square. The TRNDN with delayed feedback successfully classified 100% of the waveforms.

Optimising N and T with Delayed Feedback

To find the optimal parameters N and T, the waveform discrimination task was repeated for different N-T combinations. The NRMSE values and classification scores are presented as colour maps in Figure A.2 (a) and (b) respectively.

Figure A.2 (a) shows excellent performance, denoted by low NRMSE values (<0.05) over a wide range of N for $T \geq 2\theta$. This is similar to the result using the MGO as the NDN

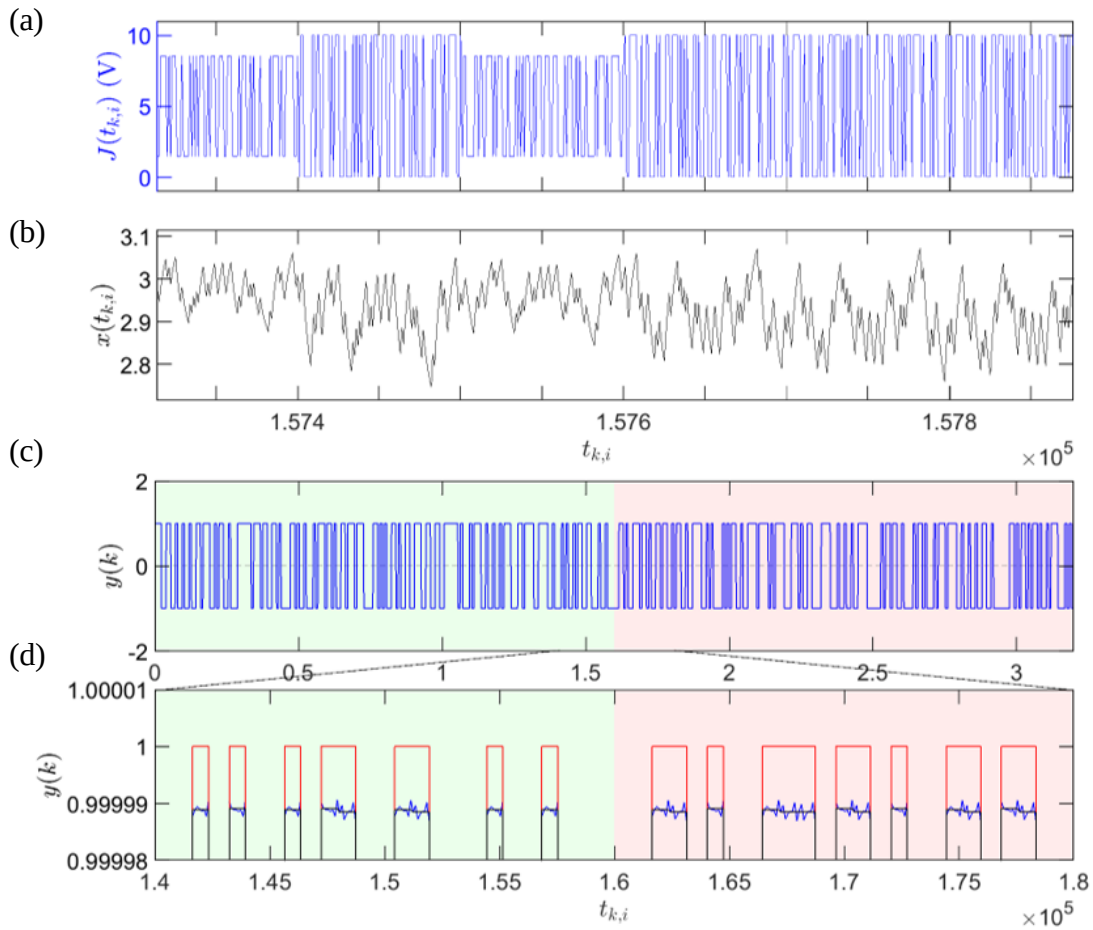


Figure A.1: Example result of the waveform discrimination task using the TRNDN *with* delayed feedback. (a) A subsection of the input sequence $J(t)$. (b) The response of the TRNDN ($N = 100$) to the stimulus shown in (a) using a response time $T = 5\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 5.74×10^{-6} and 5.62×10^{-6} for the training and testing data respectively. The error is so tiny that y is completely obscured by $\hat{y}$, so a zoomed region of the plot is shown in (d) where variations in $\hat{y}$ can be observed. The black line represents $\langle\hat{y}\rangle$: the average value of $\hat{y}$ over each waveform period.

in Figure 5.4 (a), though interestingly the best result in that case was found at $T = 1\theta$ but here $T < 2\theta$ gives a very large error (NRMSE > 0.45).

Figure A.2 (b) shows that 100% correct waveform classification was achieved even for the N-T combinations which produced a higher NRMSE, so long as $\langle\hat{y}\rangle$ has the correct sign (e.g. $N = 10$, $T = 80\theta$). The same trend was observed when using the MGO, as shown in Figure 5.4 (b). However, when the NRMSE becomes too large, $\langle\hat{y}\rangle$ may have the wrong sign for some waveforms and the classification fails. An example of failed classification is the case of $T = 1\theta$ where the classification score is ~50% for all N, denoted by the dark blue strip at the bottom of Figure A.2 (b).

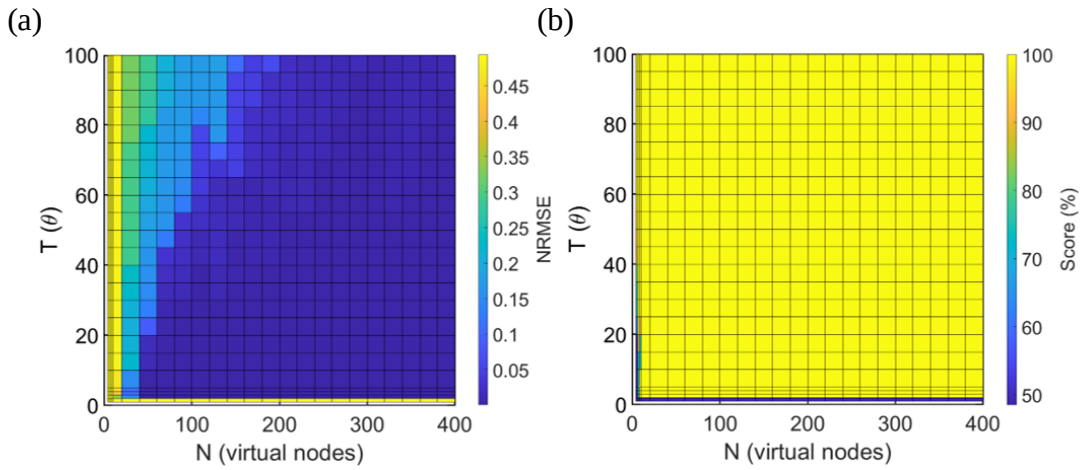


Figure A.2: Optimising *N* and *T* for the waveform discrimination task using the TRNDN *with* delayed feedback. (a) The NRMSE and (b) the % classification scores, as a function of N and T for the TRNDN *with delayed feedback*. The lowest NRMSE 6.97×10^{-7} was found at $N, T = 200, 4\theta$.

The optimal performance for the TRNDN with delayed feedback was found at $N = 200$, $T = 4\theta$ which resulted in a NRMSE of 6.97×10^{-7} and a classification score of 100%. Because the optimal result is visually equivalent to that shown in Figure A.1, the optimal result is not shown in detail.

A.1.2 Without Delayed Feedback

Example Result without Delayed Feedback

Figure A.3 presents an example result of the waveform discrimination task using the TRNDN *without* delayed feedback. Figure A.3 (a) shows a subsection of the input sequence $J(t)$ and Figure A.3 (b) shows the corresponding response of the TRNDN i.e. the explanatory variable x. Figure A.3 (c) contains the target function y (red) and virtual reservoir output $\hat{y}$ (blue) for the entire sequence of 400 waveforms. The TRNDN without delayed feedback produced a NRMSE between y and $\hat{y}$ of ~0.3, with the error arising from large fluctuations in $\hat{y}$. To show these fluctuations in more detail, a zoomed region of Figure A.3 (c) is presented in Figure A.3 (d). The black line in Figure A.1 (d) is $\langle\hat{y}\rangle$: the average of $\hat{y}$ over each waveform cycle (8 time steps τ). $\langle\hat{y}\rangle$ is used to conduct the final classification by classifying waveforms with positive $\langle\hat{y}\rangle$ as sine and negative $\langle\hat{y}\rangle$ as square. Despite the large NRMSE, the TRNDN without delayed feedback was still able to successfully classify 100% of the waveforms.

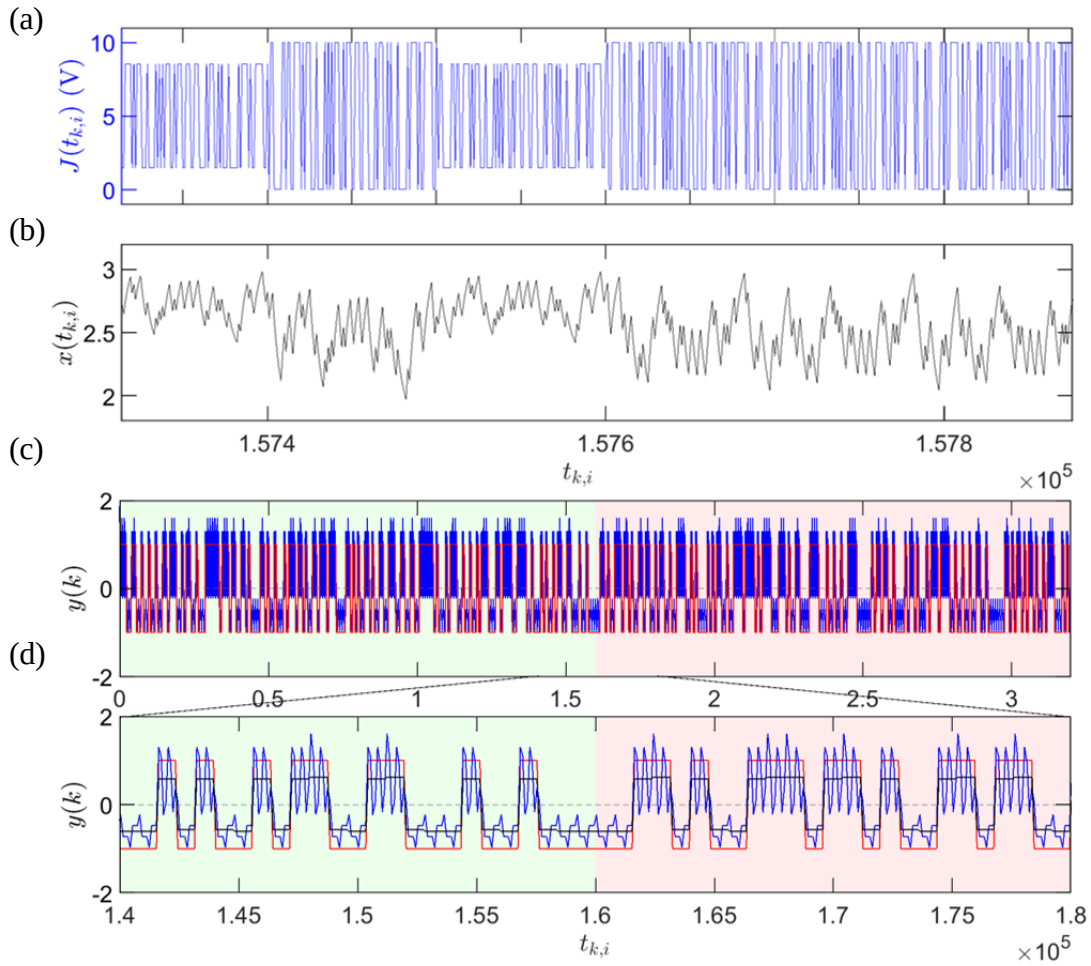


Figure A.3: Example result of the waveform discrimination task using the TRNDN *without* delayed feedback. (a) A subsection of the input sequence $J(t)$. (b) The response of the TRNDN ($N = 100$) to the stimulus shown in (a) using a response time $T = 5\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.319 and 0.315 for the training and testing data respectively. (d) A zoomed region of (c) where fluctuations in $\hat{y}$ can be seen more closely. The black line represents $\langle \hat{y} \rangle$: the average value of $\hat{y}$ over each waveform period.

Optimising N and T without Delayed Feedback

The waveform discrimination task was again repeated for different N-T combinations, but this time using the TRNDN without delayed feedback. The NRMSE values and classification scores are presented as colour maps in Figure A.4 (a) and (b) respectively. The NRMSE values (Figure A.4 (a)) follow the same trend as in the case of the MGO (Figure 5.6 (a)): there is a dark band representing optimal performance which lies along $T \sim N$. This dark band is both qualitatively and quantitatively similar to the MGO case. The classification scores for the TRNDN without delayed feedback is 100% for all N-T combinations (Figure A.4 (b)). Again, this demonstrates that high classification scores can be achieved for this task, despite

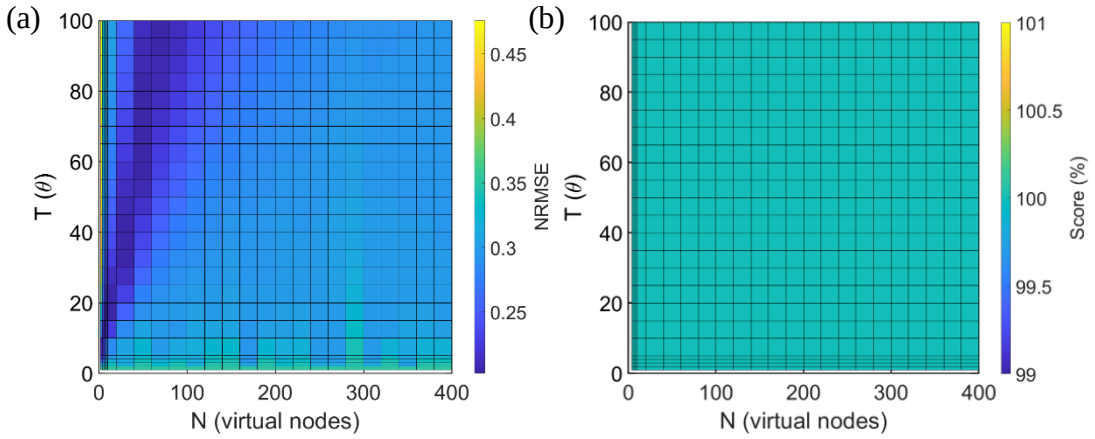


Figure A.4: Optimising N and T for the waveform discrimination task using the TRNDN *without* delayed feedback. (a) The NRMSE and (b) the % classification scores, as a function of N and T for the TRNDN *without delayed feedback*. The lowest NRMSE was found at $N, T = 40$, 70θ.

a large NRMSE, so long as $\langle \hat{y} \rangle$ has the correct sign. The optimal result was found for $N = 40$ and $T = 70\theta$ which produced a classification score of 100% and a NRMSE of 0.202. This result is shown in Figure A.5.

The difference between the results from the TRNDN with and without delayed feedback (compare Figure A.2 (a) and Figure A.4 (a)) is very similar to what was observed from the MGO (compare Figure 5.4 (a) and Figure 5.6 (a)). When delayed feedback is included, the virtual reservoir has recurrent connectivity and good performance can be obtained from a wide range of N-T combinations providing that the time constant is no longer than approximately half of the delay line (i.e. $T < N\theta/2 = \tau/2$). If the time constant is longer than $\tau/2$ and delayed feedback is included, then the virtual reservoir has too much memory and its state begins to depend on the sequence of examples, rather than the sequence of input values within each example. However, when the delayed feedback is removed, the virtual reservoir has much less memory due to the lack of direct recurrence. In this case, a longer time constant becomes beneficial in providing some indirect recurrence to the reservoir, allowing for its previous state to influence its current state. Of course, if the time constant is too long, the virtual reservoir state will again depend on the random sequence of examples and the higher-dimensional representation may be different for instances of the same class, depending on what class came before it. This is why the dark band in Figure A.4 (a) lies along $T \sim N$: time constants which are about the same length as the delay line ($\tau = N\theta$) provide the optimum balance between too much memory and too little memory.

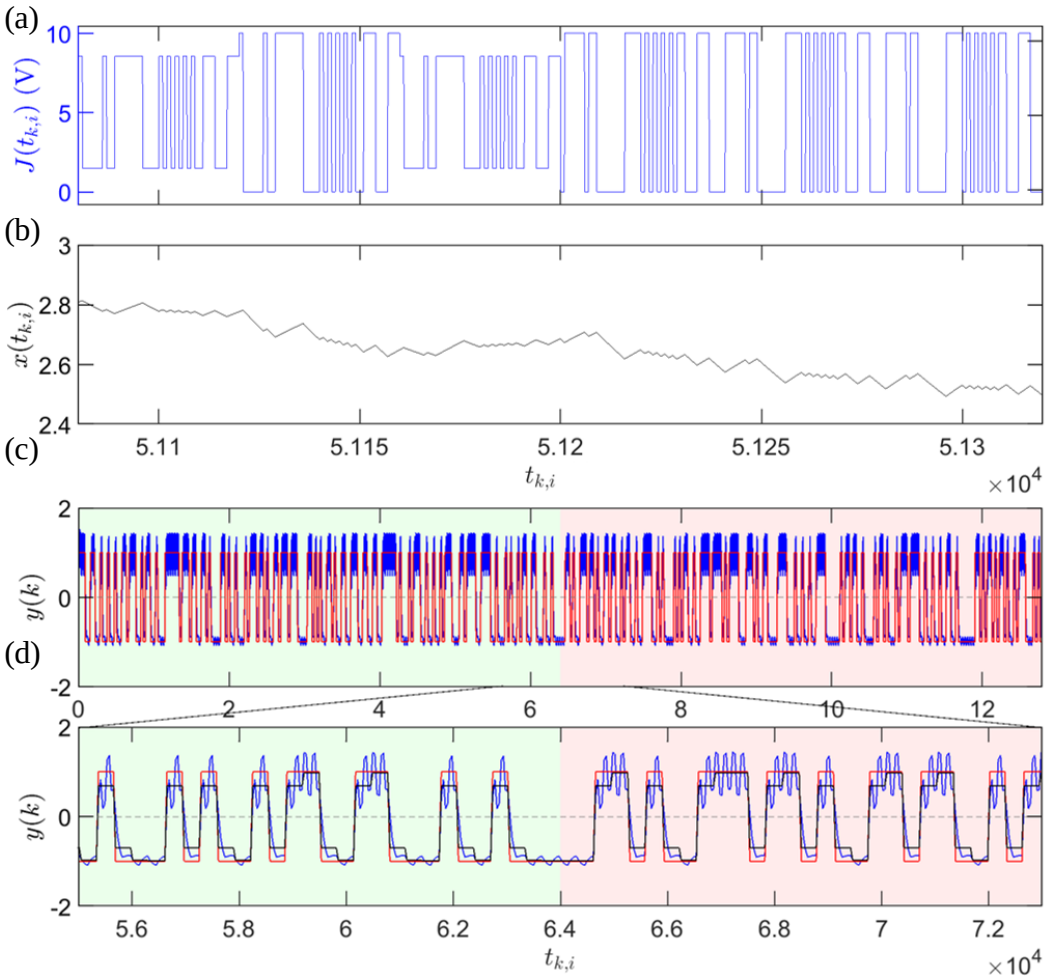


Figure A.5: Optimal result of the waveform discrimination task using the TRNDN *without* delayed feedback. (a) A subsection of the input sequence $J(t)$. (b) The response of the TRNDN ($N = 40$) to the stimulus shown in (a) using a response time $T = 70\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.207 and 0.202 for the training and testing data respectively. (d) A zoomed region of (c) where fluctuations in $\hat{y}$ can be seen more closely. The black line represents $\langle \hat{y} \rangle$: the average value of $\hat{y}$ over each waveform period.

A.2 NARMA10

A.2.1 With Delayed Feedback

Example Result with Delayed Feedback

Figure A.6 presents an example result of the NARMA10 task using the TRNDN *with* delayed feedback. Figure A.6 (a) contains a subsection of the input sequence $J(t)$ showing 12 time steps τ corresponding to 12 random input values $u(k)$. Figure A.6 (b) shows the corresponding response of the TRNDN to the input shown in Figure A.6 (a). Figure A.6 (c) presents the target function y (red) and virtual reservoir output $\hat{y}$ (blue) for the entire

sequence of 1000 input values $u(k)$. A zoomed region of Figure A.6 (c) is shown in Figure A.6 (d) which allows the difference between $\hat{y}$ and y to be seen clearly. This difference is characterised by NRMSE values of 0.099 for the training data and 0.215 for the testing data.

Optimising N and T with Delayed Feedback

To find the optimal parameters N and T, the NARMA10 task was repeated for different N-T combinations. The NRMSE values as a function of N and T are presented as a colour map in Figure A.7 (a). Compared to the MGO case (Figure 5.9 (a) where the best NRMSE values

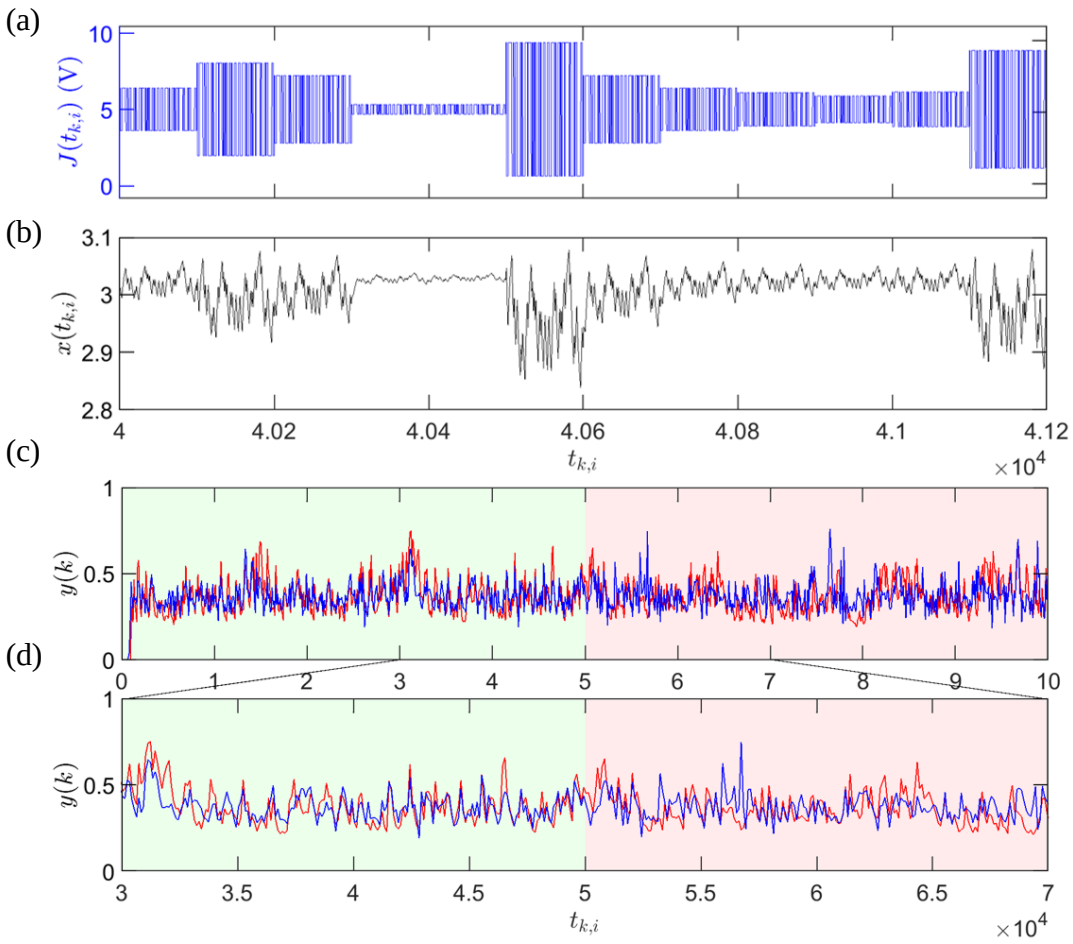


Figure A.6: Example result of the NARMA10 task using the TRNDN *with* delayed feedback. (a) A subsection of the input sequence $J(t)$ showing 12 time steps τ corresponding to 12 of the random input values $u(k)$. (b) The response of the TRNDN ($N = 100$) to the stimulus shown in (a) using a response time $T = 5\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.099 and 0.215 for the training and testing data respectively. (d) A zoomed region of (c) where fluctuations in $\hat{y}$ can be seen more closely.

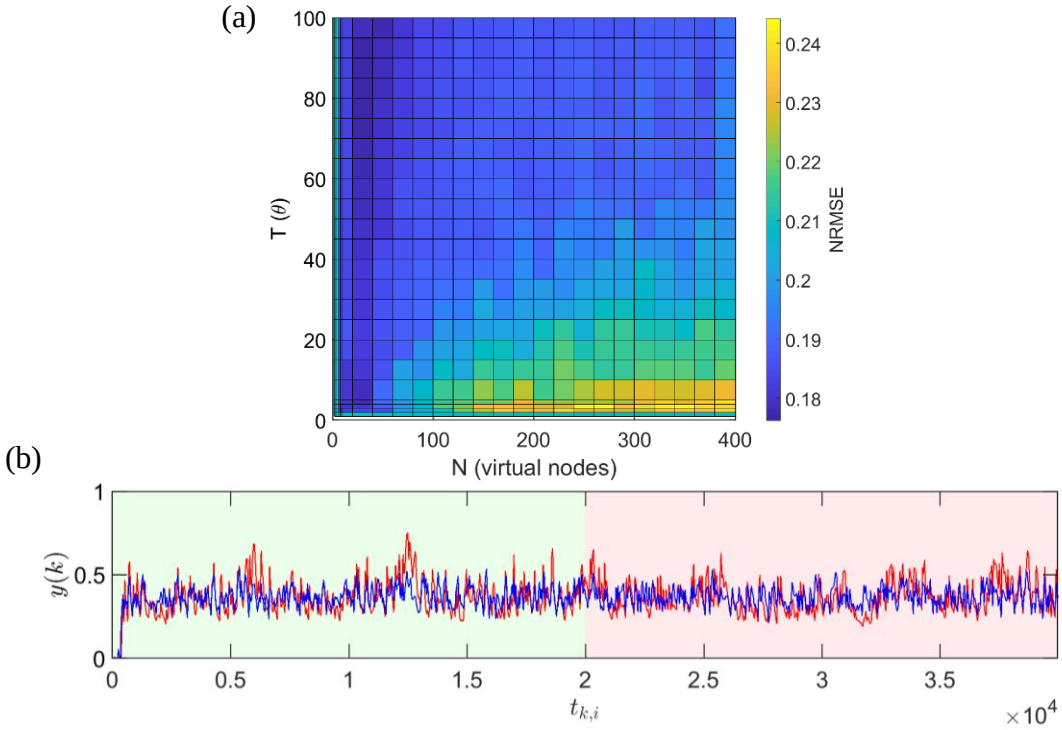


Figure A.7: Optimising N and T for the NARMA10 task using the TRNDN *with* delayed feedback. (a) The testing NRMSE as a function of N and T for the TRNDN with delayed feedback. (b) The optimal result from (a) was found at $(N, T = 20, 90\theta)$ where the virtual reservoir output $\hat{y}$ (blue) approximates the NARMA10 target function y (red) with the minimum NRMSE of 0.176.

were ~0.05, the TRNDN performs significantly worse with the lowest NRMSE being 0.176 for $N, T = 20, 90\theta$, the result of which is shown in Figure A.7 (b).

The trend across the N-T plane is also different: the MGO case in Figure 5.9 (a) features a narrow horizontal dark region indicating optimal performance for low T over a wide range of N, but in the TRNDN case the dark band is a vertical strip signalling optimal performance for low N over a wide range of T. This same feature does appear faintly in Figure 5.9 (a) as a light green band, but it is washed out by the much stronger trend of lower NRMSE towards small T.

Appendix A.1.1 showed that the TRNDN with delayed feedback could perform the waveform discrimination task just as well as the MGO with delayed feedback. Thus, it is curious that for the NARMA10 task, the TRNDN performs so much worse. It is possible that the TRNDN model may be optimised to perform better at this particular task. For example, the performance of the TRNDN could be explored as a function of the delayed feedback strength parameter κ in Eq. (53), though this is not investigated in this thesis.

A.2.2 Without Delayed Feedback

Example Result without Delayed Feedback

Figure A.8 presents an example result of the NARMA10 task using the TRNDN *without* delayed feedback. Figure A.8 (a) contains a subsection of the input sequence $J(t)$ showing 12 time steps τ corresponding to 12 random input values $u(k)$. Figure A.8 (b) shows the corresponding response of the TRNDN to the input shown Figure A.8 (a). Figure A.8 (c) presents the target function y (red) and virtual reservoir output $\hat{y}$ (blue) for the entire sequence of 1000 input values $u(k)$. A zoomed region of Figure A.6 (c) is shown in Figure

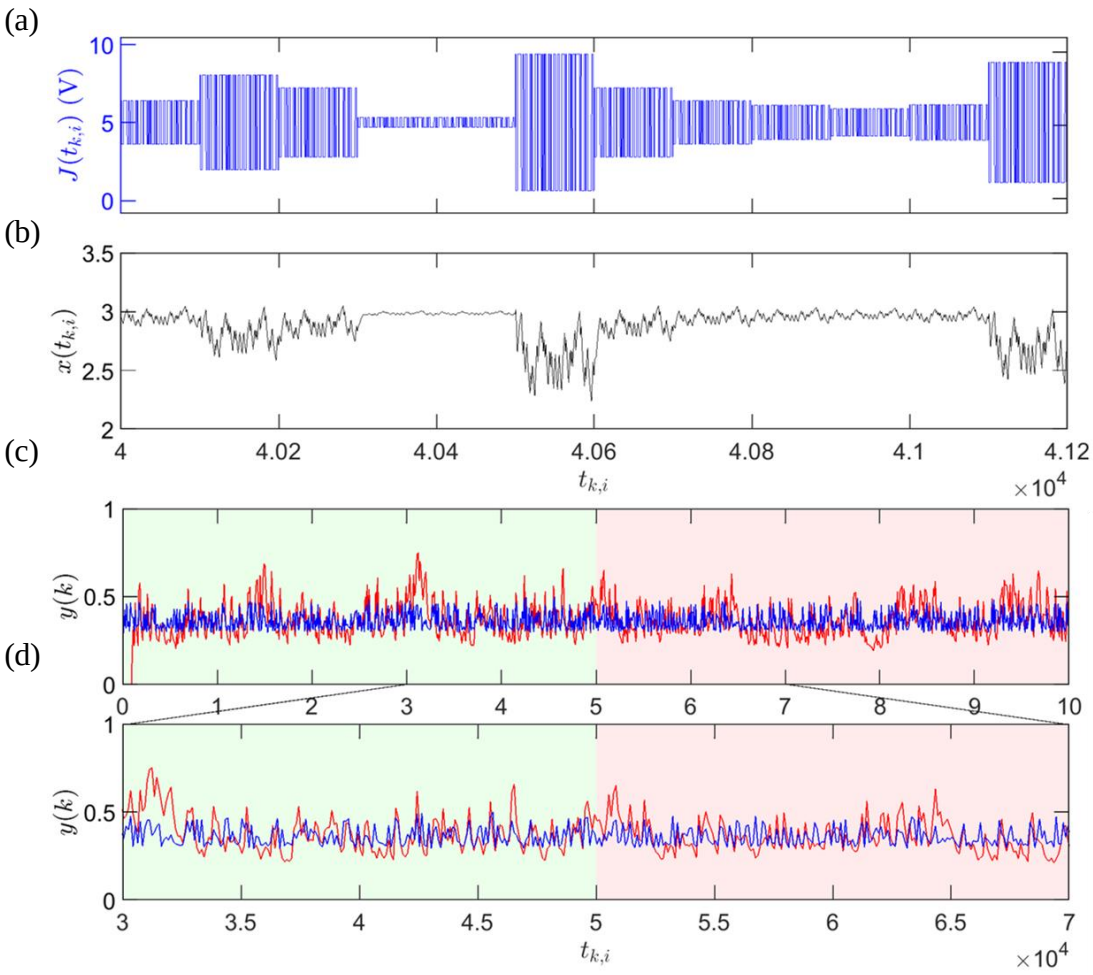


Figure A.8: Example result of the NARMA10 task using the TRNDN *without* delayed feedback. (a) A subsection of the input sequence $J(t)$ showing 12 time steps τ corresponding to 12 of the random input values $u(k)$. (b) The response of the TRNDN ($N = 100$) to the stimulus shown in (a) using a response time $T = 5\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.134 and 0.187 for the training and testing data respectively. (d) A zoomed region of (c) where fluctuations in $\hat{y}$ can be seen more closely.

A.6 (d) which allows the difference between $\hat{y}$ and y to be seen clearly. This difference is characterised by NRMSE values of 0.134 for the training data and 0.187 for the testing data.

Optimising N and T without Delayed Feedback

Again, the NARMA10 task was repeated for different N-T combinations to find the optimal parameters for the TRNDN without delayed feedback. The NRMSE values as a function of N and T are presented as a colour map in Figure A.9 (a). The trend across the N-T plane is very similar to the MGO case without delayed feedback (Figure 5.11 (a)) which featured a dark band signalling optimal performance for $T \sim 10N\theta = 10\tau$. However, in Figure A.9 (a), the dark band is broken into vertical strips, with some values of N showing the same trend but much weaker. For example, for $N = 18, 20$, the performance is strongly optimised for $T \sim 180, 200$, consistent with the dark band at $T \sim 10\tau$. However, for $N = 22$-32, the variation in NRMSE as a function of T is much weaker. There is no obvious pattern to which values of N shows the strong T dependence. Thus, these variations are attributed to the difference between the masks used for each N. The masks were generated randomly in accordance with

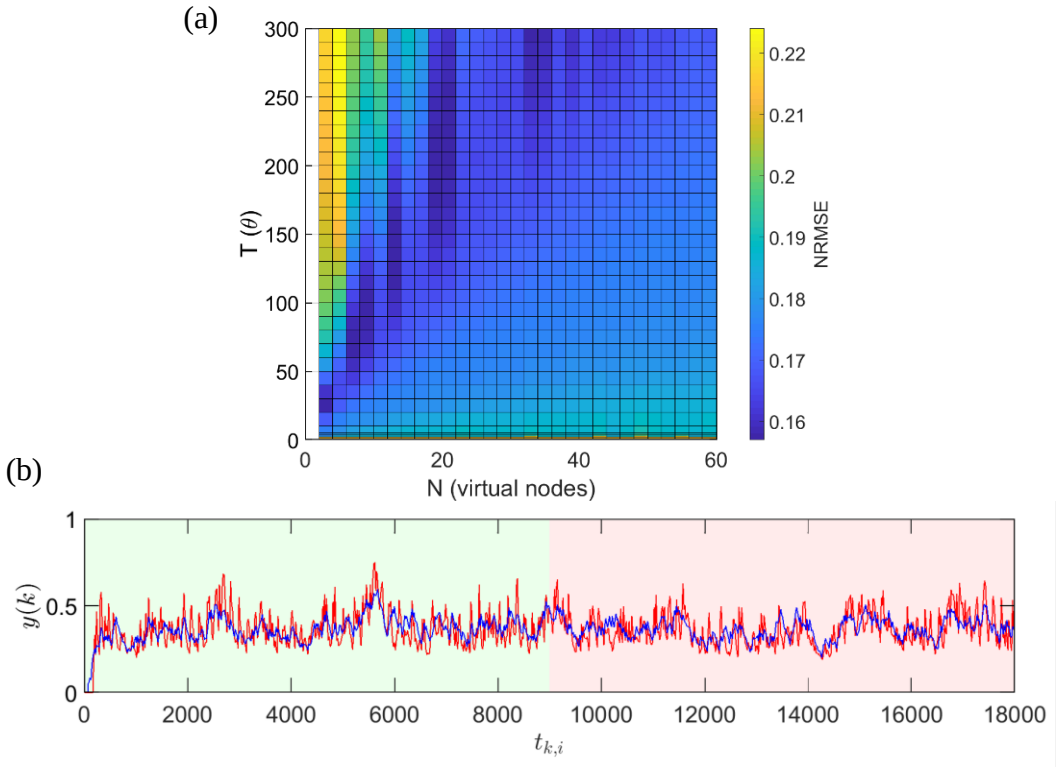


Figure A.9: Optimising N and T for the NARMA10 task using the TRNDN *without* delayed feedback. (a) The test NRMSE as a function of N and T for the TRNDN without delayed feedback. (b) The optimal result from (a) was found at $(N, T = 18, 180\theta)$ where the virtual reservoir output $\hat{y}$ (blue) approximates the NARMA10 target function y (red) with the minimum NRMSE of 0.157.

Ref. [205], though Ref. [230] shows that the masks can be optimised and that using random masks can lead to variations in performance, especially for small N. However, it remains a curious fact that the same variations were not observed for the MGO in Figure 5.11 (a) where the exact same mask sequences were used.

The optimal result from Figure A.9 (a) was found at $N, T = 18, 180\theta$ which produced a NRMSE of 0.157. This result is strikingly similar to the optimal result from the MGO without delayed feedback in Figure 5.11 (b). Again, the time constant is equivalent to ten time steps ($T = 10\tau$), thus providing the virtual reservoir with sufficient memory to predict each value of the target waveform based on the ten preceding values. A consequence of the long time constant is that high frequency components of $\hat{y}$ are damped, but the low frequency variations in $\hat{y}$ fit the target function very well.

A.3 Summary of TRNDN Results

The TRNDN model has been shown to be capable of performing waveform classification and time series prediction. The TRNDN performed the waveform discrimination task just as well as the MGO when feedback was included, and slightly better when feedback was not included. In terms of the NRMSE between the reservoir output $\hat{y}$ and the target signal y, the trend across the N-T plane was found to be both qualitatively and quantitatively similar to the MGO when delayed feedback was not included. This trend presents as a dark band representing optimal performance for $T \sim N\theta$, illustrating that long time constants (relative to θ) can provide recurrence to the virtual reservoir. However, due to the output squashing (i.e. using $\langle \hat{y} \rangle$ to classify each waveform), the percentage of correct classifications was found to be 100% almost independently of variations in the NRMSE.

The TRNDN did not perform as well as the MGO at the NARMA10 task when delayed feedback was included. The trend across the N-T plane was also very different to that of the MGO, the best case having 3-4 times the error as compared with the best result from the MGO. However the optimal result from the TRNDN with delayed feedback (NRMSE ~ 0.18) is comparable to that of [205] and achieving similar results in the experimental domain would still be a significant achievement.

The results of the NARMA10 task without delayed feedback were found to be substantially the same as the MGO. The trend across the N-T plane showed a dark band

signifying optimal performance for $T{\sim}10N$, again demonstrating that in the absence of delayed feedback, recurrence can be facilitated by a sufficiently long time constant.

Appendix B - Numerical SRNDN Results

This Appendix describes the performance of the switching regime nonlinear dynamical node model (SRNDN) for the same waveform discrimination and NARMA10 tasks considered elsewhere in the thesis.

B.1 Waveform Discrimination

B.1.1 With Delayed Feedback

Figure B.1 shows an example of the waveform discrimination task for $T = 5\theta$ and $N = 100$ with delayed feedback. Figure B.1 (a) shows a zoomed plot of the input sequence $J(t)$ and (b) shows the SRNDN response $x(t)$ i.e. the resulting switching rate. The switching rate features spikes of activity separated by periods of zero activity which occur stochastically in response to the input sequence. The N node responses are shown in different colours in Figure B.1 (c), and again in Figure B.1 (d) after being multiplied by the weights found during training. Interestingly, the optimal weights result in many of the large spikes being oppositely weighted to large spikes in other node responses which occur in the same time step τ. The resulting effect is that when the weighted node responses are summed to produce the final reservoir output $\hat{y}$, many of the large spikes cancel each other out reducing the error between $\hat{y}$ and the target function y. Figure B.1 (e) shows that $\hat{y}$ (blue) is a poor approximation of y (red) resulting in large NRMSE values (see caption). A zoomed region of (e) is shown in Figure B.1 (f), where the average reservoir output over each waveform cycle $\langle \hat{y} \rangle$ (black) can be seen fluctuating close to zero from waveform to waveform. In this case, $\langle \hat{y} \rangle$ is not a good indicator for the input waveform, resulting in a poor classification score of 50.5% which is equivalent to a random guess.

The waveform discrimination task was repeated for different combinations of N and T in order to find values that optimise the SRNDN performance. Figure B.2 (a) shows a colourmap representing the NRMSE for a wide range of N-T combinations. The NRMSE values are very large throughout the N-T plane signalling poor performance for all parameter combinations. This is reflected by the classification scores in Figure B.2 (b) which fluctuate randomly around 50% with no clear dependence on N or T.

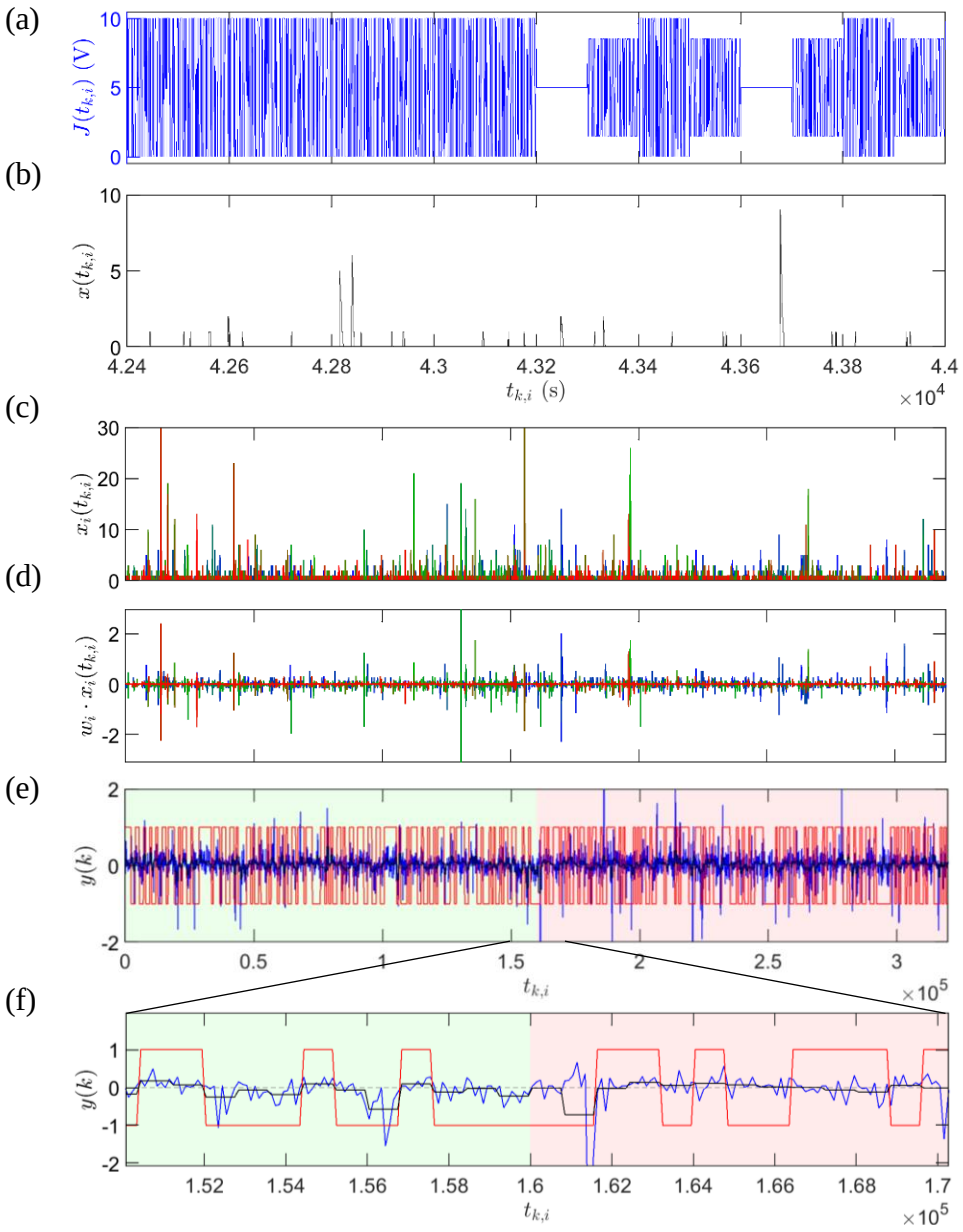


Figure B.1: Example result of the waveform discrimination task using the SRNDN *with* delayed feedback. (a) A subsection of the input sequence $J(t)$ showing a masked square and sine wave. (b) The SRNDN switching rate in response to the stimulus shown in (a) using a response time $T = 5\theta$. (c) The different virtual node responses ($N = 100$) to the entire sequence of 400 waveforms. The different colours represent different virtual nodes. (d) The virtual node responses shown in (c) after being multiplied by the weights found during the training procedure. (e) The final output of the virtual reservoir $\hat{y}$ (blue) was constructed by summing the weighted responses shown in (d). $\hat{y}$ is a poor approximation of the target function y shown in red with NRMSE values of 0.481 and 0.523 for the training and testing segments respectively. (f) A zoomed region of panel (e) showing the average reservoir output $\langle \hat{y} \rangle$ over each waveform cycle. $\langle \hat{y} \rangle$ should be > 0 for a sine classification and < 0 for a square classification. In this case, a poor classification score of 50.5% was found.

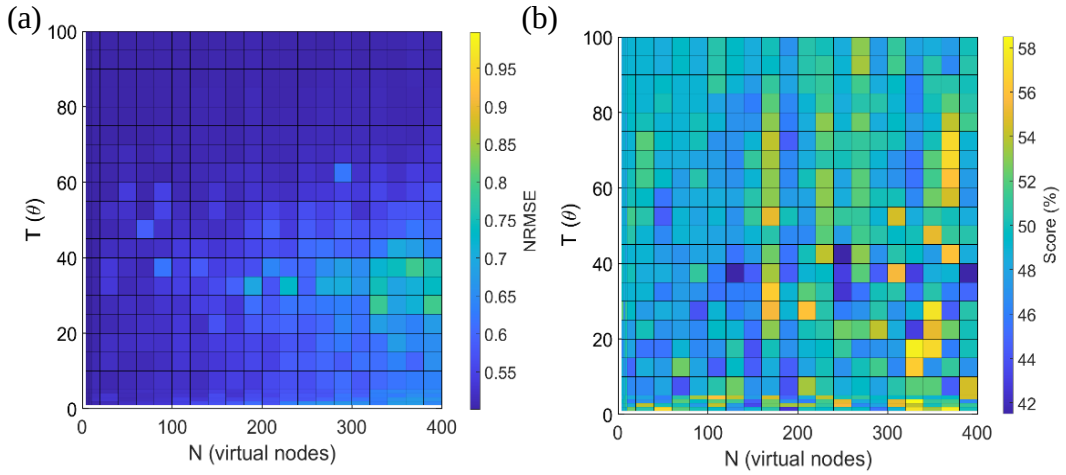


Figure B.2: Optimising N and T for the waveform discrimination task using the SRNDN *with* delayed feedback. (a) Colourmap showing the NRMSE between the reservoir output $\hat{y}$ and the target y as a function of the number of virtual nodes in the reservoir N and the response time T. The NRMSE values are very large representing poor reservoir performance. (b) The resulting classification scores as a function of N and T. The classification scores fluctuate around 50% corresponding to a random guess. There is no clear dependence on N or T.

The N-T combination which produced the lowest NRMSE was found at $N, T = 10, 50\theta$, with a NRMSE of 0.50. The result for this case (shown in Figure B.3 (d)) is trivial as it corresponds to a switching rate (Figure B.3 (a)) that features almost zero activity. A flat horizontal line given by $y = 0$ corresponds to a NRMSE of 0.5, and the virtual reservoir output $\hat{y}$ is zero for almost all time steps. This trivial result arises because long time constants (such as $T = 50$) inhibit large values in the synthetic switching rate. Eq. (59) is used to implement the response time in the SRNDN by limiting the change in switching rate from point to point i.e. $\Delta x = (x_i - x_{i-1})$, to $\Delta x/T$. Because the switching rate values are initially drawn from a power law CDF (Eq. (56)), it is very likely that $x_{i-1} = 0$. In this case, a large T makes it very likely that the next switching rate value x_i is also zero after rounding to the nearest integer. The result is a synthetic switching rate which is zero for almost every time step, thus producing the nearly flat $\hat{y}$ shown in Figure B.3 (d).

In contrast with the result shown in Figure B.3, the case shown in Figure B.1, which corresponds to a small response time ($T = 5\theta$), features a comparatively active switching rate containing many spikes and avalanches of activity (Figure B.1 (c)). However, the NRMSE produced by that case is even larger than that of the trivial result in Figure B.3 (d). The fact that the non-trivial switching rate produces a larger NRMSE than the trivial (almost flat) switching rate is strong indication that the spikes/avalanches of activity produced by the SRNDN are not good predictors of the class of input waveform.

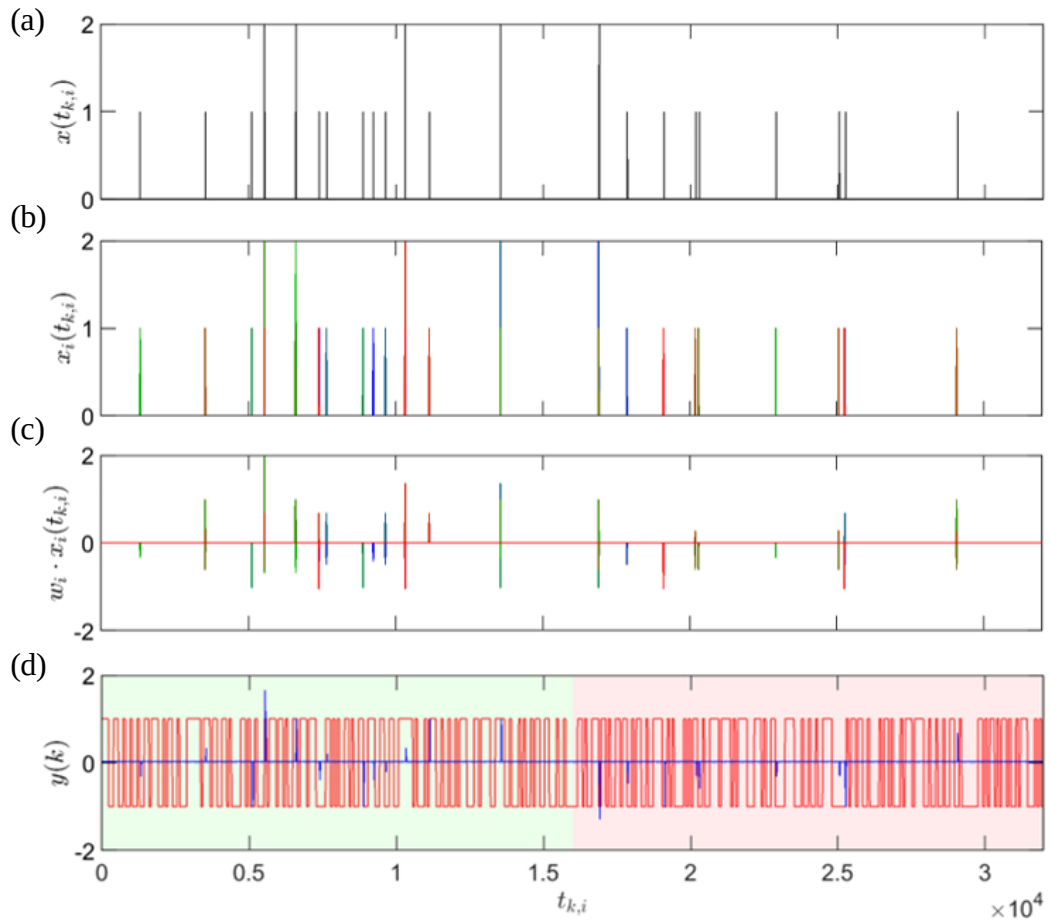


Figure B.3: Optimal result of the waveform discrimination task with the SRNDN; $N = 10, T = 50\theta$. (a) The SRNDN switching rate in response to the entire sequence of 400 waveforms using a response time $T = 50\theta$. (b) The different virtual node responses ($N = 10$). The different colours represent different virtual nodes. (c) The virtual node responses shown in (b) after being multiplied by the weights found during the training procedure. (d) The final output of the virtual reservoir $\hat{y}$ (blue) was constructed by summing the weighted responses shown in (d). $\hat{y}$ is a poor approximation of the target function y shown in red with NRMSE values of 0.498 and 0.500 for the training and testing segments respectively. In this case, a poor classification score of 51.5% was found, equivalent to a random guess.

The poor performance of the SRNDN may be partly due to the probabilistic nature of the synthetic switching rate, whereby switching rate values are drawn from the CDF using random numbers. It may also be partly due to the large number of switching rate values which are zero: a consequence of drawing rate values from a power law distribution. Both of these possibilities are explored in Appendix B.3.2.

B.1.2 Without Delayed Feedback

The waveform discrimination task was performed using the SRNDN without the delayed feedback mechanism, as described by Eq. (60). Figure B.4 shows the result for $T = 5\theta$ and $N = 100$. The result is characterised by a NRMSE of 0.530 and a classification score of 52.5%, both of which signify poor task performance. Compared to the same case with delayed feedback (Figure B.1), the SRNDN response (Figure B.4 (a)) and the resulting approximation of the target function (Figure B.4 (e)) are qualitatively the same. That is to say that the inclusion/exclusion of the delayed feedback mechanism appears to make very little difference to the performance of the SRNDN.

Again, the waveform discrimination task was repeated for different combinations of N and T. Figure B.5 (a) shows a colourmap representing the NRMSE for a wide range of N-T combinations, while Figure B.5 (b) shows a colourmap representing the corresponding classification scores. The results in Figure B.5 (a) are very similar the case with delayed feedback (Figure B.2 (a)), with large NRMSE values across the N-T plane, the lowest being 0.5 which is trivial as it is equivalent to a flat line at $y = 0$. The classification scores in Figure B.5 (b) however, do show some improvement over the case with delayed feedback (Figure B.2 (b)) in the form of a pocket of good performance (75-80%) in the lower right quadrant of the colourmap. These results correspond to a small response time and a large number of virtual nodes.

The improvement in task performance from removing the delayed feedback mechanism suggests that the delayed feedback mechanism is ineffective for the SRNDN. The ineffectiveness of the delayed feedback mechanism is surprising given that, when using the MGO and the TRNDN, the inclusion of the delayed feedback mechanism had a profoundly positive effect on the task performance. The problem may therefore be that the SRNDN itself is not able to exploit the recurrence provided by the delayed feedback mechanism. Again, this may be related to the stochasticity of the SRNDN response, or to the large number of zeros in the synthetic switching rate, possibilities which are further explored in Appendix B.3.2.

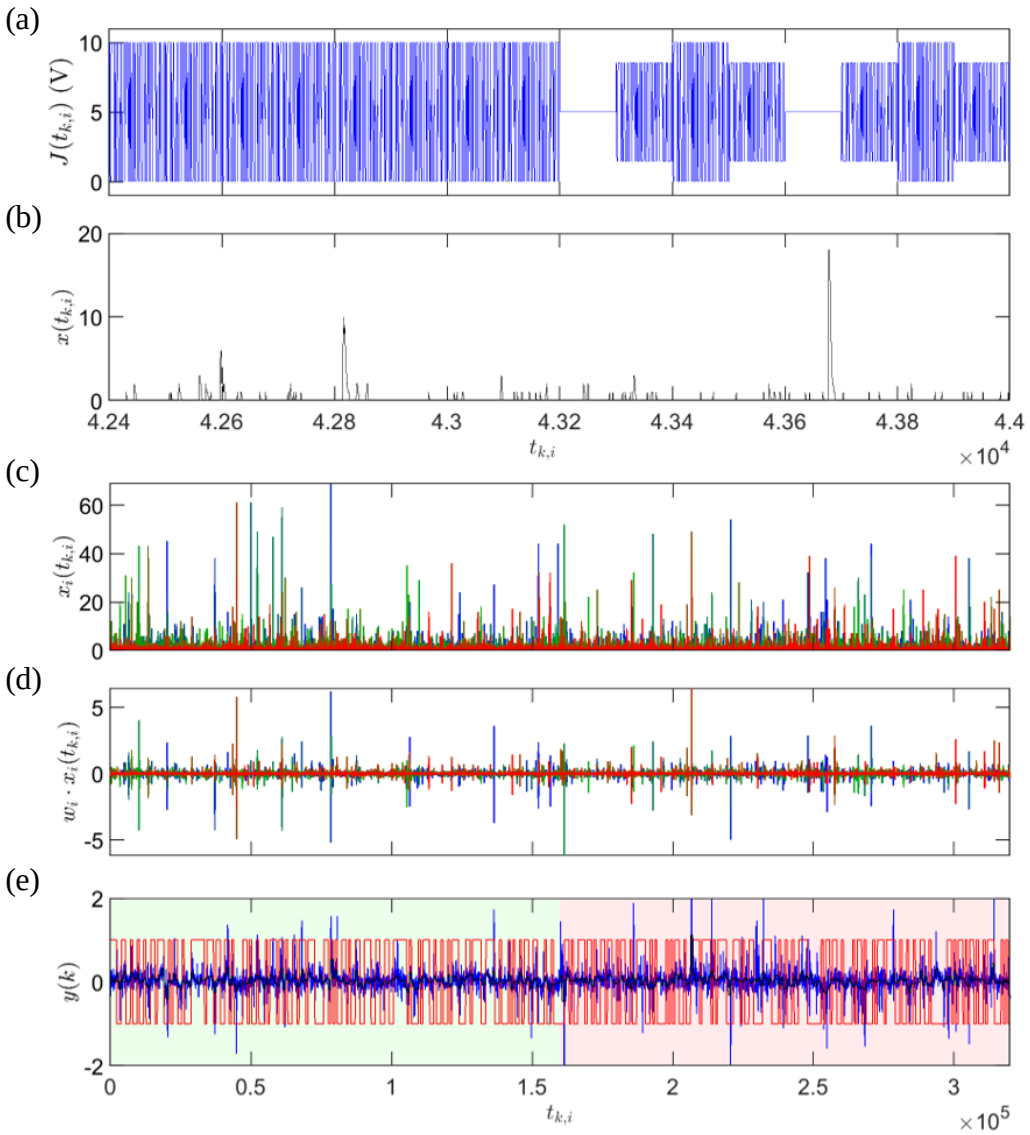


Figure B.4: Waveform discrimination task using SRNDN without delayed feedback; $N = 100, T = 5\theta$. (a) A subsection of the input sequence $J(t)$ showing a masked square and sine wave. (b) The SRNDN switching rate in response to the stimulus shown in (a) using a response time $T = 5\theta$. (c) The different virtual node responses ($N = 100$) to the entire sequence of 400 waveforms. The different colours represent different virtual nodes. (d) The virtual node responses shown in (c) after being multiplied by the weights found during the training procedure. (e) The final output of the virtual reservoir $\hat{y}$ (blue) was constructed by summing the weighted responses shown in (d). $\hat{y}$ is a poor approximation of the target function y shown in red with NRMSE values of 0.482 and 0.530 for the training and testing segments respectively. (f) A zoomed region of panel (e) showing the average reservoir output $\langle \hat{y} \rangle$ over each waveform cycle. $\langle \hat{y} \rangle$ should be > 0 for a sine classification and < 0 for a square classification. In this case, a poor classification score of 52.5% was found.

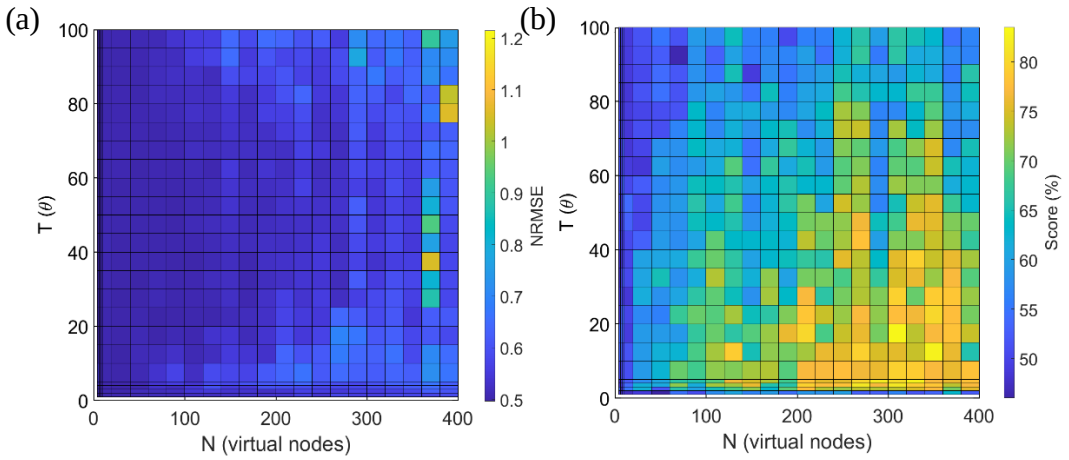


Figure B.5: Waveform discrimination task using SRNDN without feedback. (a) Colourmap showing the NRMSE between the reservoir output $\hat{y}$ and the target y as a function of the number of virtual nodes in the reservoir N and the response time T. The NRMSE values are very large representing poor reservoir performance. (b) The resulting classification scores as a function of N and T. The classification scores are an improvement on those in Figure B.2 (b), featuring a region of good performance (75-80%) in the lower right quadrant of the colourmap corresponding to small T and large N.

B.2 NARMA10

B.2.1 With Delayed Feedback

Figure B.6 shows an example of the NARMA10 task for $T = 5\theta$ and $N = 100$ with delayed feedback. Figure B.6 (a) shows a subsection of the input sequence $J(t)$ corresponding to ten time steps i.e. ten of the random input values $u(k)$ which are encoded in the amplitude of $J(t)$. Figure B.6 (b) shows the resulting switching rate featuring the spikes of activity and periods of quiescence typical of the SRNDN. Figure B.6 (c) contains the responses of all 100 virtual nodes to the full sequence of 1000 input values and Figure B.6 (d) shows those same responses after having been multiplied by the weights found in during the training procedure. Finally, the reservoir output $\hat{y}$ and the NARMA10 target function y are shown in Figure B.6 (e). The performance is characterised by a NRMSE of 0.135 for the training data and 0.263 for the testing data. This result, while not as good as that of the MGO or the TRNDN, is still comparable to results from Refs. [205] and [238] which are given in Table 5.

It is of particular interest that the testing NRMSE is approximately twice as large as the training NRMSE. Recall that previous results (e.g. NARMA10 with MGO in Figure 5.9 (b)) showed small differences between the NRMSE of the training and testing data which are typical, but significant differences such as the one observed in Figure B.6 may be a sign of

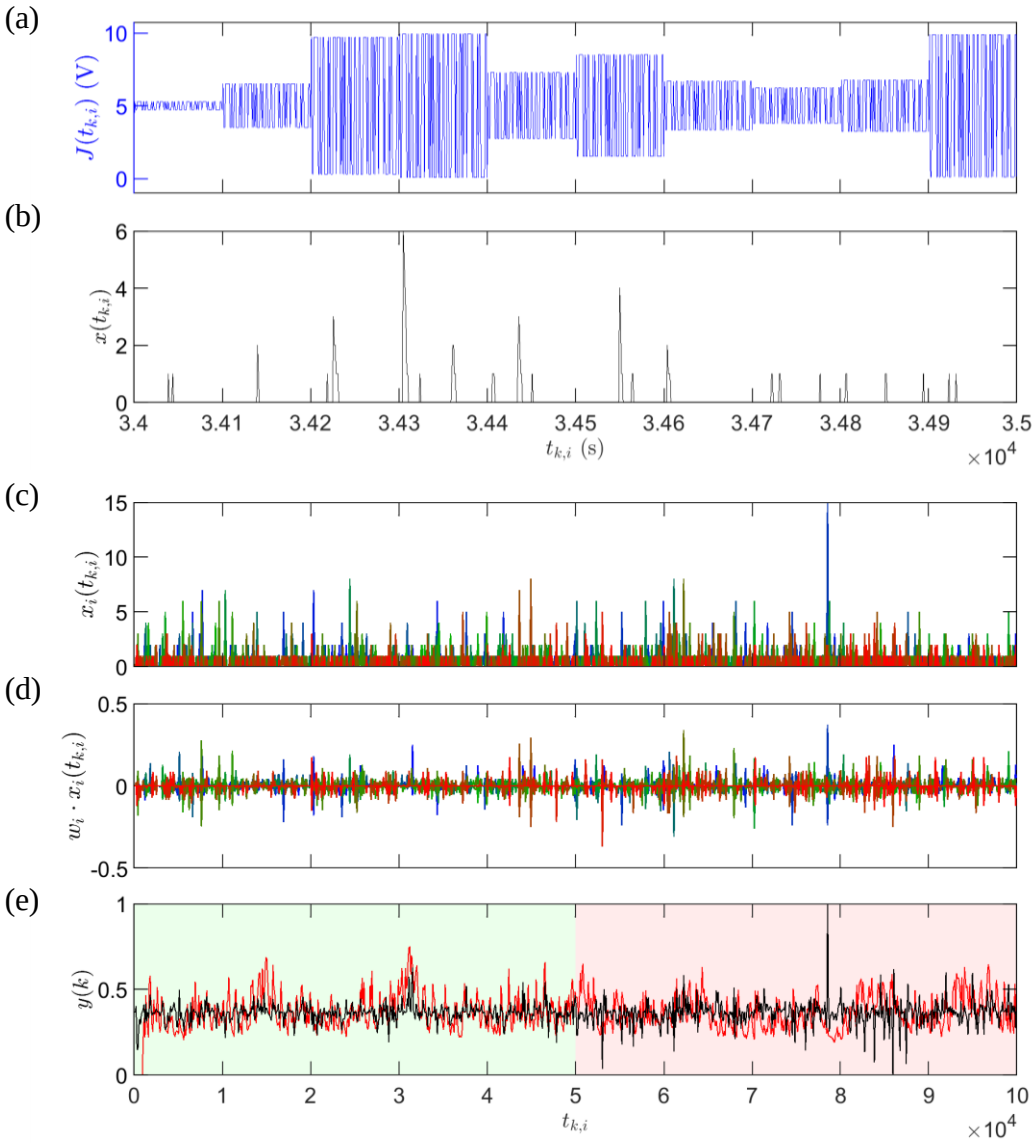


Figure B.6: NARMA10 task using SRNDN with delayed feedback; $N = 100, T = 5\theta$. (a) A subsection of the input sequence $J(t)$ showing ten masked input values $u(k)$. (b) The SRNDN switching rate in response to the stimulus shown in (a) using a response time $T = 5\theta$. (c) The different virtual node responses ($N = 100$) to the entire sequence of 1000 input values. The different colours represent different virtual nodes. (d) The virtual node responses shown in (c) after being multiplied by the weights found during the training procedure. (e) The final output of the virtual reservoir $\hat{y}$ (black) was constructed by summing the weighted responses shown in (d). The result is characterised by NRMSE values of 0.135 and 0.263 for the training and testing data respectively. The difference between the training and testing NRMSE values is significant, indicating that the training procedure may be overfitting the data (Section 4.3.5).

overfitting (see Section 4.3.5).

The NARMA10 task was then repeated for different N-T combinations. Figure B.7 contains colourmaps representing (a) the training NRMSE, and (b) the testing NRMSE, each as a function of N and T. Figure B.7 (a) shows a clear trend of better performance for larger N and smaller T, while Figure B.7 (b) shows the opposite trend of worse performance for larger N and smaller T. The fact that as performance improves for the training data, performance concurrently deteriorates for the testing data strongly suggests that overfitting is taking place.

Recall that overfitting occurs when the weights are trained so that $\hat{y}$ fits too closely to the training data and therefore fails to generalize to the testing data which the reservoir was not specifically trained on [51,52]. In this case, regularization methods such as Tikhonov regularization, more commonly known as ridge regression, can be used to mitigate the effect of overfitting [248].

The largest difference between the training and testing NRMSE from Figure B.7 is for $N, T = 400, 1\theta$. The results for this case are shown in Figure B.8. It is clear from Figure B.8 (e) that the virtual reservoir output $\hat{y}$ is a good approximation of the target function y for the

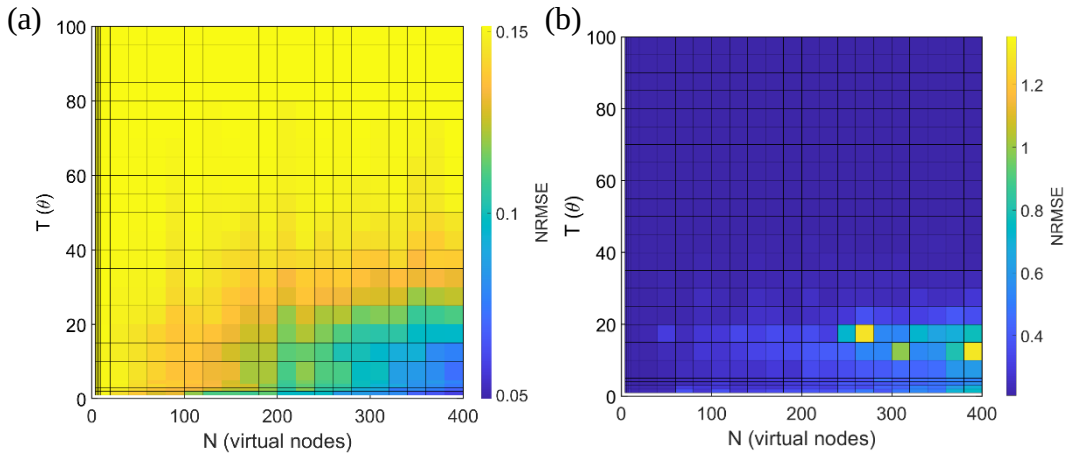


Figure B.7: Optimising N and T for the NARMA10 task using SRNDN *with* feedback. Colourmaps showing the NRMSE between the reservoir output $\hat{y}$ and the target y as a function of N and T for (a) the training data, and (b) the testing data. (a) represents good training performance with NRMSE as low as 0.05 for $N, T = 400, 1\theta$. There is a clear trend of better performance for larger N and smaller T. (b) represents poor performance during testing for all combinations of N and T, with the opposite trend to that in (a): performance decreases with increasing N and decreasing T. The complementary trends observed between the training and testing data are a strong indication of overfitting.

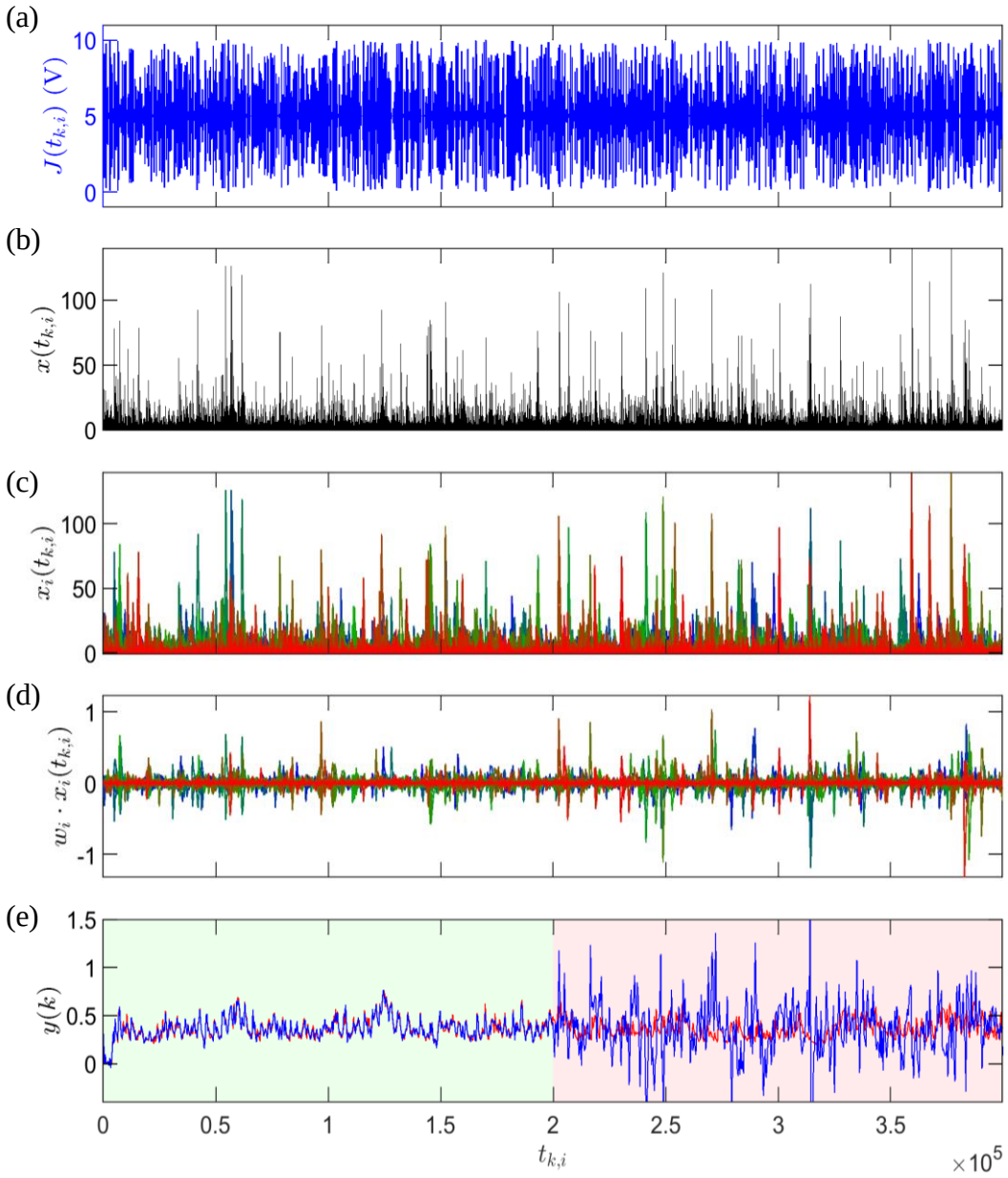


Figure B.8: NARMA10 task using SRNDN with delayed feedback; $N = 400, T = 1\theta$. (a) The full input sequence $J(t)$ showing 1000 masked input values $u(k)$. (b) The SRNDN switching rate in response to the stimulus shown in (a) using a response time $T = 1\theta$. (c) The different virtual node responses ($N = 400$). The different colours represent different virtual nodes. (d) The virtual node responses shown in (c) after being multiplied by the weights found during the training procedure. (e) The final output of the virtual reservoir $\hat{y}$ (blue) was constructed by summing the weighted responses shown in (d). The result is characterised by NRMSE values of 0.048 and 0.655 for the training and testing data respectively. The difference between the training and testing NRMSE values is significant, confirming that overfitting is taking place.

training data (first half) which has a low NRMSE of 0.048, but not for the testing data (second half), which has a NRMSE of 0.655. This result confirms that the data is being overfit.

B.2.2 Overfitting and Ridge Regression

The poor testing performance for the NARMA10 task may be improved by using ridge regression. Section 4.3.5 outlines the procedure for training the weights by ridge regression, which necessitates a validation step in order to select the optimal ridge parameter. When training the weights using Eq. (47), the data is divided into two segments: 50% for training and 50% for testing. However, when using ridge regression, the data must be divided into three segments: 50% for training, 30% for validation, and 20% for testing. The training data is used to find optimal sets of weights for a range of ridge parameter values λ. Then the optimal λ is determined by using each set of weights to fit the validation data and choosing the weights which produce the lowest NRMSE. The set of weights found using the optimal λ are then used to construct the target function, and the performance is assessed using the testing data.

In this thesis, a recursive search was used to find the optimal ridge parameter, which allows a wide range of ridge parameter values to be assessed efficiently. The recursive search involves selecting ten ridge parameter values which are spread over a wide range; finding the value from this range by using the set of weights it produces to assess the validation data; repeating the process over a narrower range centred around the optimal value. The search range is refined four times to find the optimal ridge parameter and the corresponding optimal set of weights.

The NARMA10 task was repeated for $N, T = 400, 1\theta$ using ridge regression to train the weights, instead of Eq. (47). The results are presented in Figure B.9. Because ridge regression only modifies the training of the weights, the input sequence $J(t)$, switching rate x and virtual node responses X_i are all identical to those shown in Figure B.8 (a-c). Figure B.9 (a) shows the results of the recursive search used to optimise the ridge parameter. Each of the four panels represents a finer search, resulting in an optimal ridge parameter $\lambda = 288.86$. Figure B.9 (b) shows the virtual node responses after being multiplied by the weights corresponding to the optimal ridge parameter. The amplitudes are smaller than in Figure B.8 (d) because ridge regression has the effect of reducing the magnitude of the weights. Figure B.9 (c) shows the virtual reservoir output $\hat{y}$ and target function y. The pale green, blue and red backgrounds represent the training, validation and testing data respectively. The performance is

characterised by NRMSE values of 0.172 for the training data, 0.362 for the validation data and 0.453 for the testing data.

The ridge regression did have the desired effect of reducing the overfitting. The training performance decreased and the testing performance improved, as can be seen by comparing the bottom panels of Figure B.8 and Figure B.9 (note the different y-scales). However, the improved testing NRMSE of 0.453 is still much larger than those of the best performances

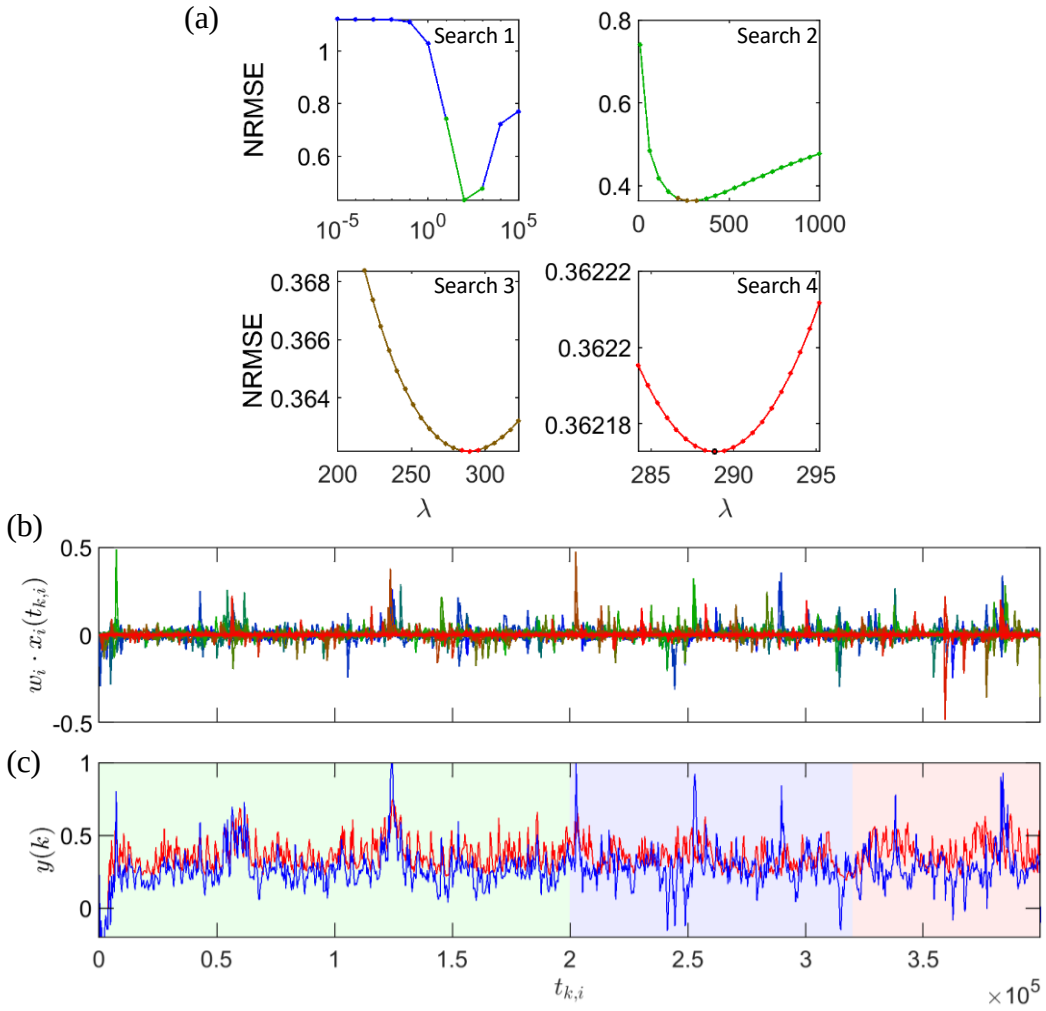


Figure B.9: NARMA10 task using ridge regression; $N = 400, T = 1\theta$ with delayed feedback. (a) The recursive search used to find the optimal ridge parameter. Successive searches are performed over narrower ranges to find the minimum NRMSE for the validation data (b) The virtual node responses after being multiplied by the weights found during the training procedure. (c) The final output of the virtual reservoir $\hat{y}$ (blue) was constructed by summing the weighted responses shown in (b). The result is characterised by NRMSE values of 0.172 for the training data (green background), 0.362 for the validation data (blue background), and 0.453 for the testing data (red background). The difference between the training and testing NRMSE values is significantly reduced by training with ridge regression, but the testing performance remains poor.

of other NDN models (e.g. MGO, TRNDN) and of the reported NRMSE values from the literature given in Table 5. Hence, while the ridge regression did reduce the overfitting, the SRNDN performance at NARMA10 remains poor.

B.2.3 Without Delayed Feedback

For completeness, the NARMA10 task was performed using the SRNDN without the delayed feedback component. Figure B.10 shows the result for the case $N, T = 100$, 5θ trained using linear regression i.e. Eq. (47). The result is substantially the same as the case with delayed feedback shown in Figure B.6, demonstrating that, as in the waveform discrimination task, the delayed feedback mechanism makes little difference to the performance of the SRNDN.

To confirm that the delayed feedback mechanism is ineffective for all N-T parameter combinations, the NARMA10 task was repeated without feedback for a range of N and T. Figure B.11 contains colourmaps representing (a) the training NRMSE for training via linear regression, (b) the testing NRMSE for training via linear regression, (c) the training NRMSE for training via ridge regression, (d) the testing NRMSE for training via ridge regression.

Comparing Figure B.11 (a) and (b) with Figure B.7 (a) and (b) shows that the delayed feedback has minimal effect on the SRNDN performance at the NARMA10 task. The trends are the same with/without delayed feedback, with good training performance for large N and small T and poor testing performance for all N-T combinations. The testing NRMSE values are slightly smaller with delayed feedback (Figure B.7 (b)) but even those NRMSE values $\sim$0.25 represent poor performance, again signalling that the SRNDN may not be utilizing the recurrence provided by the delayed feedback.

Figure B.11 (c) and (d) show the training and testing NRMSEs when training via ridge regression. Comparison with Figure B.11 (a) and (b) show that the ridge regression successfully improved the testing performance at the cost of reduced training performance, as expected. Training by ridge regression also results in Figure B.11 (c) and (d) showing similar trends in NRMSE as a function of N and T. However, despite the improved testing performance when training via ridge regression, the SRNDN still performs badly at the NARMA10 task. Appendix B.3.2 explores several reasons for the inefficacy of the SRNDN.

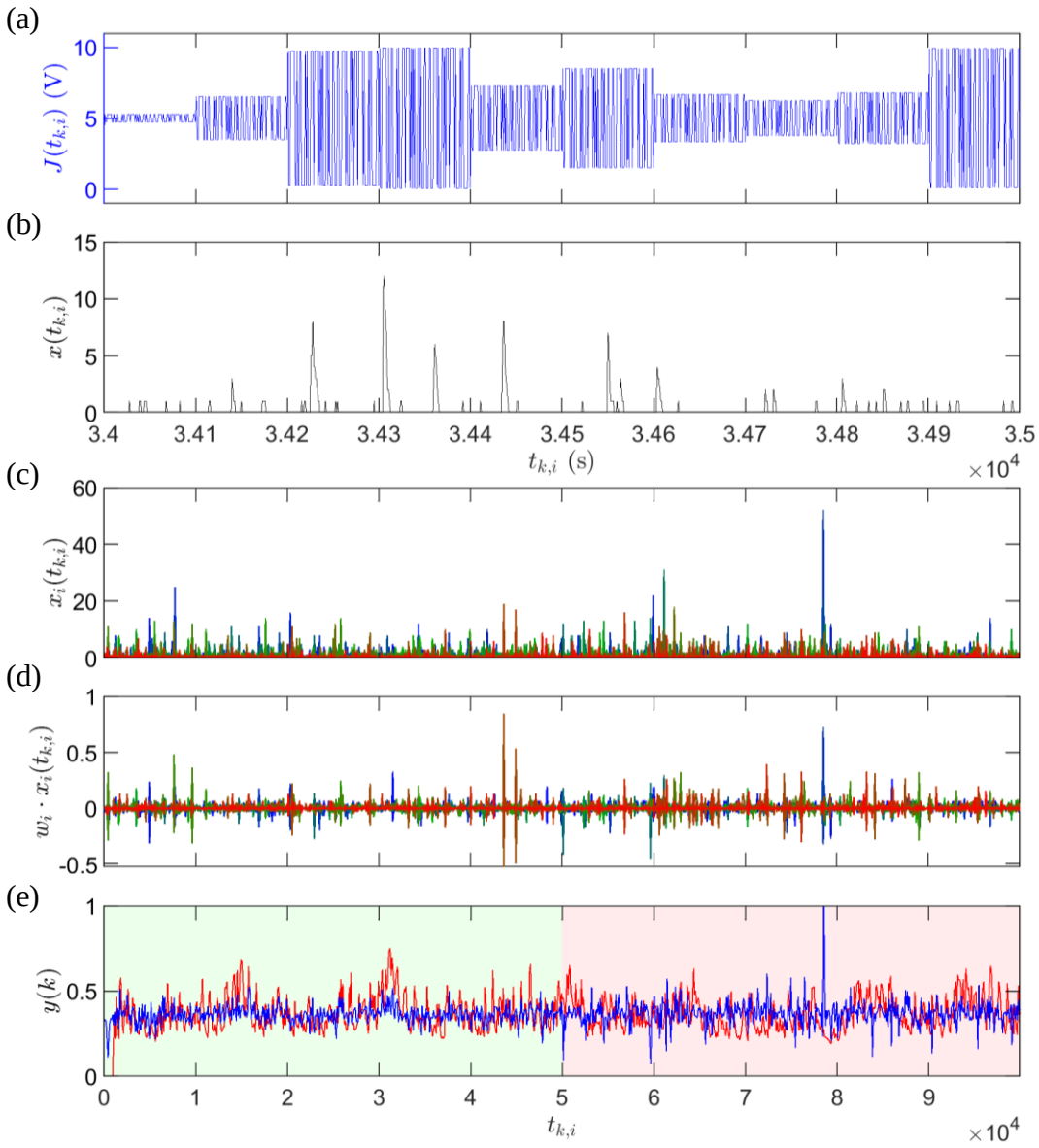


Figure B.10: NARMA10 task using SRNDN without delayed feedback; $N = 100, T = 5\theta$. (a) A subsection of the input sequence $J(t)$ showing ten masked input values $u(k)$. (b) The SRNDN switching rate in response to the stimulus shown in (a) using a response time $T = 5\theta$. (c) The different virtual node responses ($N = 100$) to the entire sequence of 1000 input values. The different colours represent different virtual nodes. (d) The virtual node responses shown in (c) after being multiplied by the weights found during the training procedure. (e) The final output of the virtual reservoir $\hat{y}$ (blue) was constructed by summing the weighted responses shown in (d). The result is characterised by NRMSE values of 0.135 and 0.261 for the training and testing data respectively. Again, the difference between the training and testing NRMSE values is significant, indicating that the training procedure is overfitting the data.

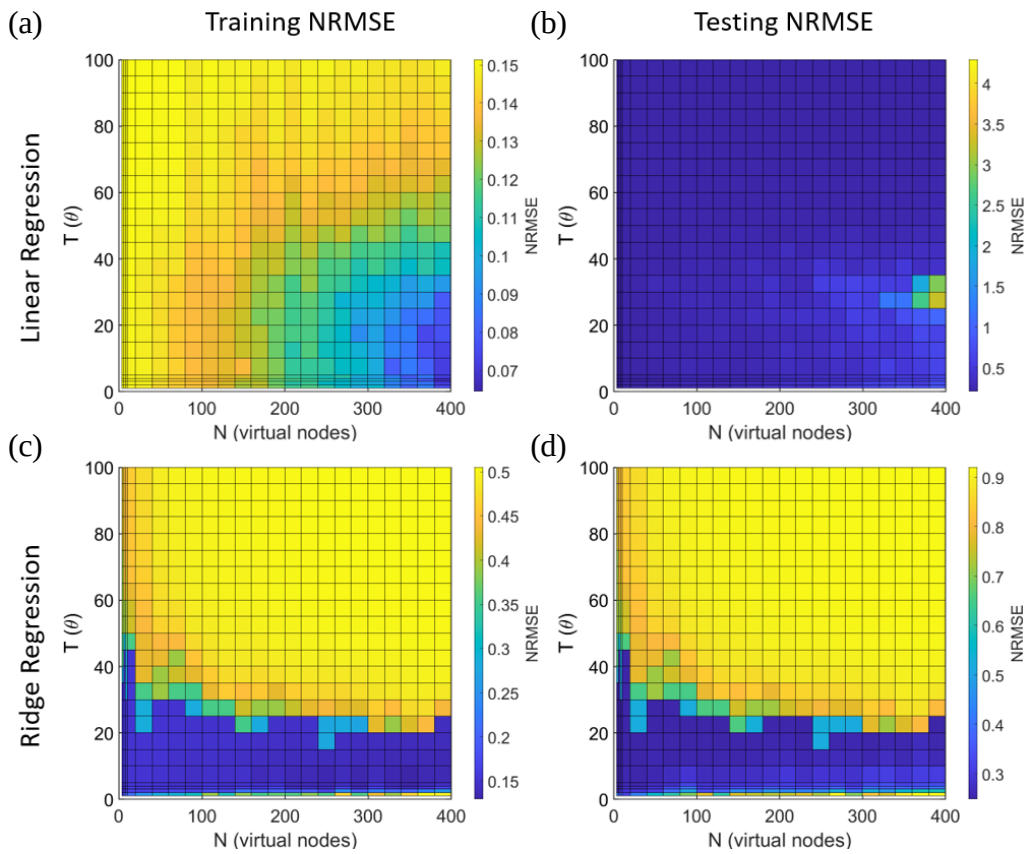


Figure B.11: NARMA10 task using SRNDN with feedback. Colourmaps showing the NRMSE between the reservoir output $\hat{y}$ and the target y as a function of N and T for (a) the *training* data with weights trained via *linear* regression, (b) the *testing* data with weights trained via *linear* regression, (c) the *training* data with weights trained via *ridge* regression, (d) the *testing* data with weights trained via *ridge* regression. When training via linear regression there is clear overfitting indicated by the good training performance and correspondingly poor testing performance. Training via ridge regression reduces the overfitting so that the training and testing performance follow similar trends and the difference between the training and testing NRMSE is reduced for each N-T combination.

B.3 Zeros and Stochasticity in the Explanatory Variable

The switching rates of a PASN, and of the SRNDN model, are approximately power law distributed as shown in Figure 5.15. This has two consequences which may be relevant to the poor performance of the SRNDN in Appendices B.1 and B.2. First is that because smaller switching rate values are more likely, there is a very high probability of having switching rate

values of zero[19] i.e. time bins which contain no events. Zeros in the output of a virtual node will remain zeros after weighting and cannot contribute to the linear combination $\hat{y}$ which approximates the target function. They will however affect the choice of weights as the training algorithm tries to minimize the error between the weighted linear sum of virtual node outputs $\hat{y}$ and the target function y. Therefore, values of zero in the switching rate should have a negative impact on the performance of the SRNDN or of a physical switching regime PASN-based NDN.

The second consequence of the power law distributed switching rate is the level of stochasticity in the response of the switching rate to the input signal. Figure 5.15 shows that the switching rate distribution in response to a single DC voltage can range over several orders of magnitude. This means that repeatedly applying the same sequence of input voltages is likely to yield switching rates which differ from trial to trial, thus making it difficult to train weights which will perform well in testing.

The TRNDN from Section 5.2 was used to explore the effect of both the number of zeros, and the level of stochasticity in the explanatory variable. The aim is to determine whether the poor performance of the SRNDN can be attributed to the presence of zeros and stochasticity in the synthetic switching rate.

B.3.1 The Effect of Zeros in the Explanatory Variable

To simulate the effect of empty bins in the PASN/SRNDN switching rate (values of zero in the explanatory variable x) the TRNDN was used, but with an increasing proportion of randomly selected values in x replaced with zeros prior to training the weights. The TRNDN was used for this task because it performs the waveform discrimination task very well and perturbing a successful model by replacing random values in the explanatory variable with zero allows for any degradation in performance to be clearly seen.

The waveform discrimination task was performed for $N = 40$ over a wide range of T with between 0-50% of the values in x replaced with zeros. In Figure B.12, the NRMSE values and classification scores are presented both as functions of T and of the fraction of x values replaced by zeros. The left column in Figure B.12 shows the NRMSE values (top) and

[19] Pure continuous power law distributions do not contain any values of zero. However, the experimental switching rate histogram in Figure 5.15 (c) is a discrete truncated power law and shows that the proportion of zeros in the switching rate is consistent with the power law distribution. The synthetic switching rate of the SRNDN shares this characteristic.

classification scores (bottom) for the waveform discrimination task with delayed feedback, while the right column shows the same without delayed feedback.

With no values of x replaced with zeros (i.e. 0%), the case *with* delayed feedback (Figure B.12, left) shows the same performance as a function of T as observed for $N = 40$ in Figure A.2 (a) and (b). The NRMSE is very close to zero at $T = 3\theta$ and increases with increasing T, while 100% classification scores are observed for all values of T. Upon replacing just 0.5% of the values in x with zeros, the performance deteriorates drastically producing NRMSE values $\sim$0.5 and classification scores $\sim$50% for all values of T. The same poor performance is observed as the number of x values replaced with zeros increases to 50%.

In the case *without* delayed feedback (Figure B.12, right), the performance with no zero replacements is the same as that in Figure A.4 (a) and (b). A minimum NRMSE occurs at $T \sim 70\theta$ corresponding to the dark band in in Figure A.4 (a), while the classification score is 100% for all T. When the fraction of zero replacements increases, the performance begins to deteriorate, but much more gradually than in the case with feedback. Although the NRMSE values increase quickly with the fraction of zero replacements, classification scores $>$80% can be observed even with 2.5% of x values replaced with zeros, only dropping to $\sim$50% (equivalent to a random guess) when $>$10% of x values are replaced with zeros.

The contrast between the cases with and without delayed feedback highlights an important feature of the delayed feedback mechanism. Typically, as demonstrated by the results from the MGO (Section 5.1) and the numerical TRNDN (Section 5.2), NDNs with delayed feedback perform better than without delayed feedback. However, when random values of x are replaced with zeros, the opposite is true. The reason for this is due to the recurrence provided by the delayed feedback mechanism which makes the state of each virtual node depend on its previous state. Recurrence is advantageous when virtual node states are representative of the present input and previous node states (i.e. no values of x replaced with zeros), leading to the observation of better performance with delayed feedback. However, when some virtual node states are replaced with zeros, which contain no information of the present input or of past states, recurrence in the reservoir allows those zero states to propagate to future time steps, influencing the virtual reservoir for much longer. This causes extensive perturbation of the states of other virtual nodes which *do* carry important information, leading to a decrease in performance when the delayed feedback is included.

This realization explains the observation in Appendix B.1 that the SRNDN performs worse when the delayed feedback mechanism is included.

The switching rate of a real PASN contains many zeros with ~75% of time bins of size $\langle IEI \rangle$ containing no events. This is a reflection of the heavy tailed IEI distributions presented in Figure 3.14. Given the negative effect of the delayed feedback mechanism when the explanatory variable contains a significant proportion of zeros, delayed feedback should not be included in a physical TRNDN implementation. Even then, the bottom right panel in Figure B.12 shows that with more than ~10 % zeros, poor task performance is likely to be observed.

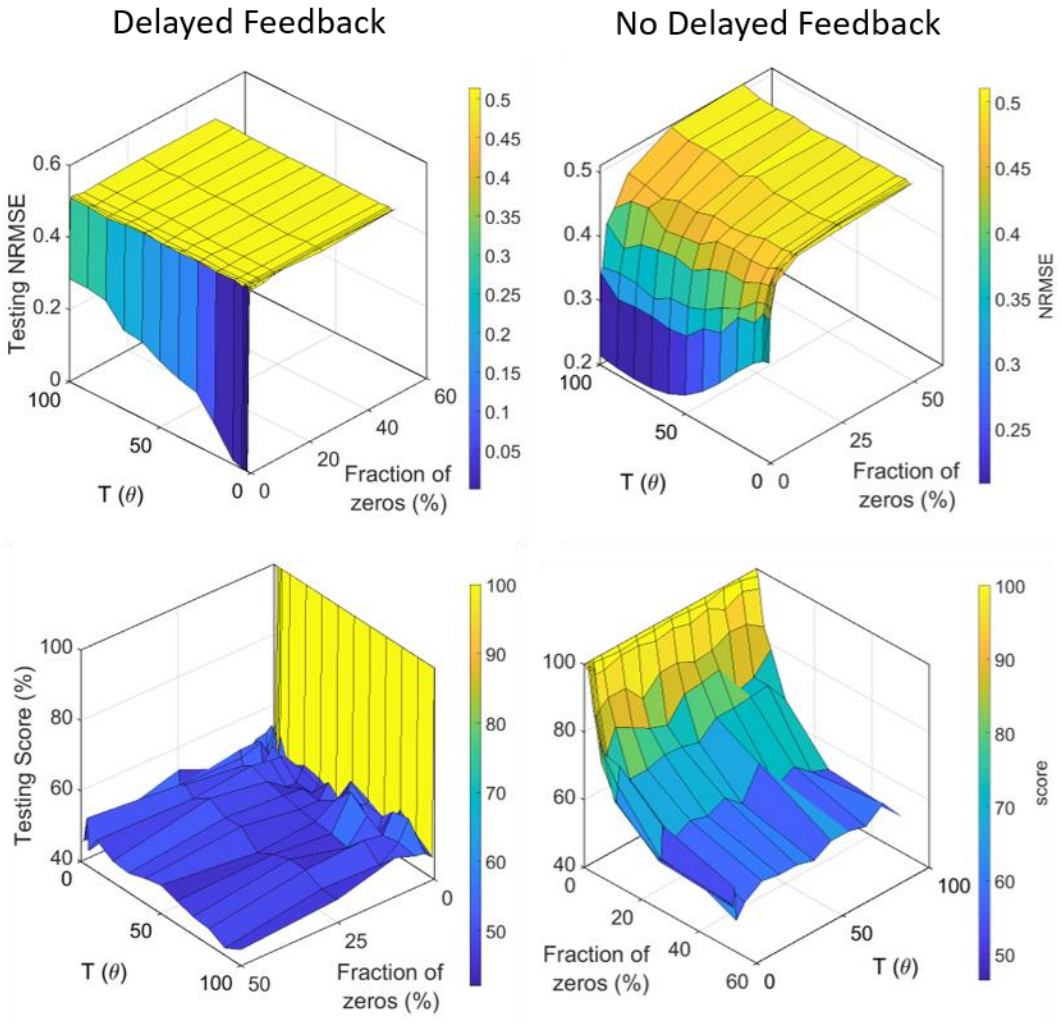


Figure B.12: The effect of zeros in the explanatory variable. Results of the waveform discrimination task using the TRNDN with random values of x replaced with zeros. The left and right columns represent cases with and without delayed feedback respectively. The top and bottom rows show the NRMSE values and classification scores each as a function of the time constant T and of the fraction of random x-values replaced with zeros. The case with delayed feedback performs poorly for any fraction of zero replacements while the case without delayed feedback shows a more gradual decrease in performance as the fraction of zero replacements increases.

B.3.2 The Effect of Stochasticity in the Explanatory Variable

To simulate the effect of stochasticity in the PASN/SRNDN switching rate the TRNDN was used again, but with an increasing level of randomness introduced to x prior to training the weights. Every value in the explanatory variable x_i was multiplied by a scaling value r_i, where r_i was drawn randomly from a normal distribution $\boldsymbol{r}$. The normal distribution has a mean of one and the standard deviation σ was used to control the level of stochasticity in x, representing the trial-to-trial variability of the switching rate. Figure B.13 (a) shows an example of three normal distributions $\boldsymbol{r}$ corresponding to different values of σ and Figure B.13 (b) shows the corresponding TRNDN responses $x(t)$ to the input sequence $J(t)$ shown in grey. If $\sigma = 0$, every value r_i equals one, and x remains unperturbed as represented by the black curve in Figure B.13 (b). As σ is increased, $\boldsymbol{r}$ becomes broader and the values of x become increasingly stochastic. The aim is to determine how much stochasticity can be tolerated by the virtual reservoir before the task performance degrades significantly.

The waveform discrimination task was performed for $N = 40$ over a wide range of T for σ in the range [0, 0.1]. In Figure B.14, the NRMSE values and classification scores are presented both as functions of T and of σ. The left column in Figure B.14 show the NRMSE values (top) and classification scores (bottom) for the waveform discrimination task with delayed feedback, while the right column shows the same without delayed feedback.

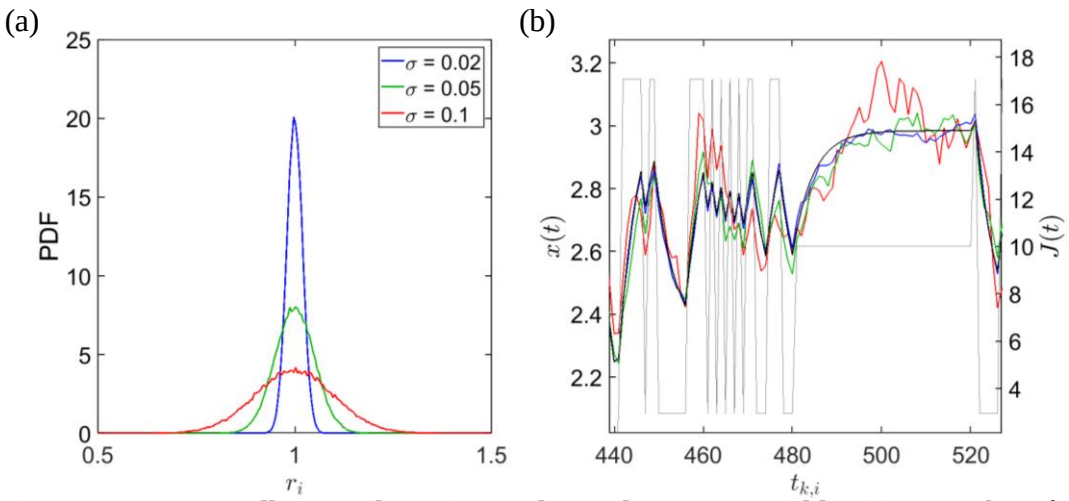


Figure B.13: Controlling stochasticity in the explanatory variable. (a) Examples of three normal distributions $\boldsymbol{r}$ with means of 1 and different standard deviations σ. (b) A sample of the TRNDN response $x(t)$ to the input sequence $J(t)$ after each value of x is scaled by values drawn randomly from $\boldsymbol{r}$. As σ increases, $\boldsymbol{r}$ becomes broader and x becomes more stochastic.

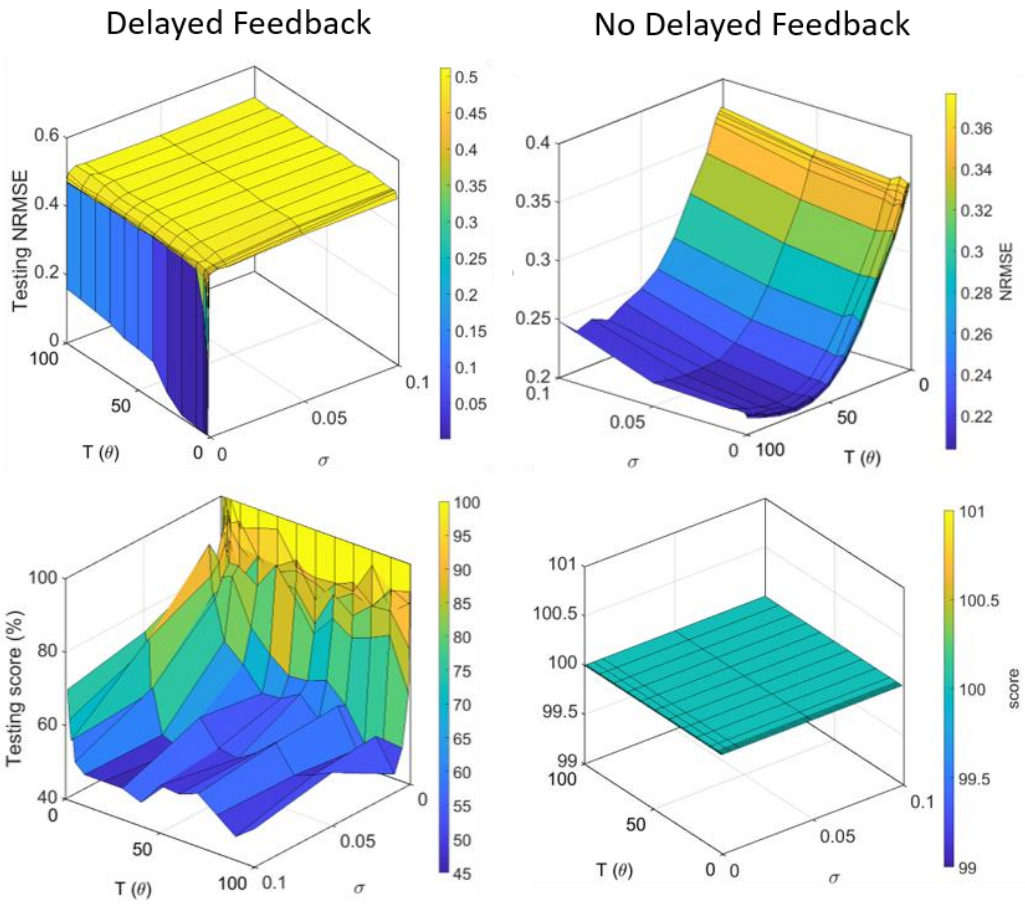


Figure B.14: The effect of stochasticity in the explanatory variable. Results of the waveform discrimination task using the TRNDN with values of x multiplied by values drawn randomly from a normal distribution $\boldsymbol{r}$ which has a mean of 1. As the standard deviation σ is increased, the values of x become increasingly stochastic. The left and right columns represent cases with and without delayed feedback respectively. The top and bottom rows show the NRMSE values and classification scores each as a function of the time constant T and of σ. The case with delayed feedback shows a significant decrease in performance with increasing σ while the case without delayed feedback appears somewhat resilient to stochastic variations.

For $\sigma = 0$, the case *with* delayed feedback (Figure B.14, left) again shows the same performance as a function of T as observed for $N = 40$ in Figure A.2 (a) and (b). The NRMSE is very close to zero at $T = 3\theta$ and increases with increasing T, while 100% classification scores are observed for all values of T. Upon increasing σ from 0 to 0.001, the performance deteriorates drastically producing NRMSE values ~0.5. The classification score shows a similar deterioration as σ increases, though there is some dependence on T with smaller T corresponding to slightly better performance.

In the case *without* delayed feedback (Figure B.14, right), the performance at $\sigma = 0$ is again the same as that in Figure A.4 (a) and (b). A minimum NRMSE occurs at $T \sim 70\theta$ corresponding to the dark band in in Figure A.4 (a). As σ is increased, the NRMSE gradually

increases, in contrast to the sudden increase in NRMSE seen in the case with feedback. The total decrease in performance with increasing σ is also significantly smaller than in the case with delayed feedback; in fact, the dependence on T is much stronger than the dependence on σ. The classification score is 100% for all values of T and σ, independent of the variation in NRMSE.

When random values of x were replaced with zeros, it was shown that the NDN with delayed feedback performed worse than the NDN without delayed feedback. The same trend was also observed when stochasticity was added to x. Again, the reason for this apparent inconsistency is the effect of the recurrence provided by the delayed feedback mechanism. When the NDN response is purely deterministic, the delayed feedback mechanism improves performance by allowing useful information to remain in the reservoir for an extended period of time. However, when the NDN response is stochastic, the recurrence also allows random variations in x to remain within the virtual reservoir, perturbing a larger number of virtual node states. Because these perturbations are different for each input waveform, a single set of weights cannot be used to construct a linear combination $\hat{y}$ which is a perfect representation of the target function y. Thus, the resulting NRMSE increases with the magnitude of the stochastic variations in x.

This appendix explored the effects of zeros and stochasticity in the explanatory variable x in order to investigate the poor performance of the SRNDN in Section 5.3. The TRNDN was used to perform the waveform discrimination task with increasing variations introduced to x in the form of both random replacements of x values with zero, and stochastic scaling. It was found in both cases that the delayed feedback mechanism causes the virtual reservoir to perform worse than when the delayed feedback is omitted. The reason is that the recurrence provided by the delayed feedback mechanism allows randomness to affect more virtual node states. As zeros and stochasticity are characteristic features of both the synthetic switching rate produced by the SRNDN and the real switching rate produced by the physical PASN in the switching regime, poor performance should be expected from these NDNs as a direct result of stochasticity. This is consistent with the generally poor performance of the numerical SRNDN in Section 5.3 which is summarized in Table 6. In conclusion, a PASN in the switching regime shows little promise of being a good choice for a NDN for TDRC because of its inherent stochasticity. If a stochastic NDN such as a PASN in the switching regime

were to be used for TDRC, the delayed feedback mechanism should be omitted for best results.

Appendix C - Physical TRNDN Results

C.1 Waveform Discrimination

This appendix contains the results of the waveform discrimination task using the physical TRNDN with the optimal time constants found in Section 6.1.1. Figure C.1 shows the result with partial delayed feedback and Figure C.2 shows the result without partial delayed feedback. These are important results to show because they represent the best performing neuromorphic implementations of real PASN devices. However, because Figure C.1 and Figure C.2 appear superficially similar to other figures in Section 6.1.1, they are included in this appendix to aid readability.

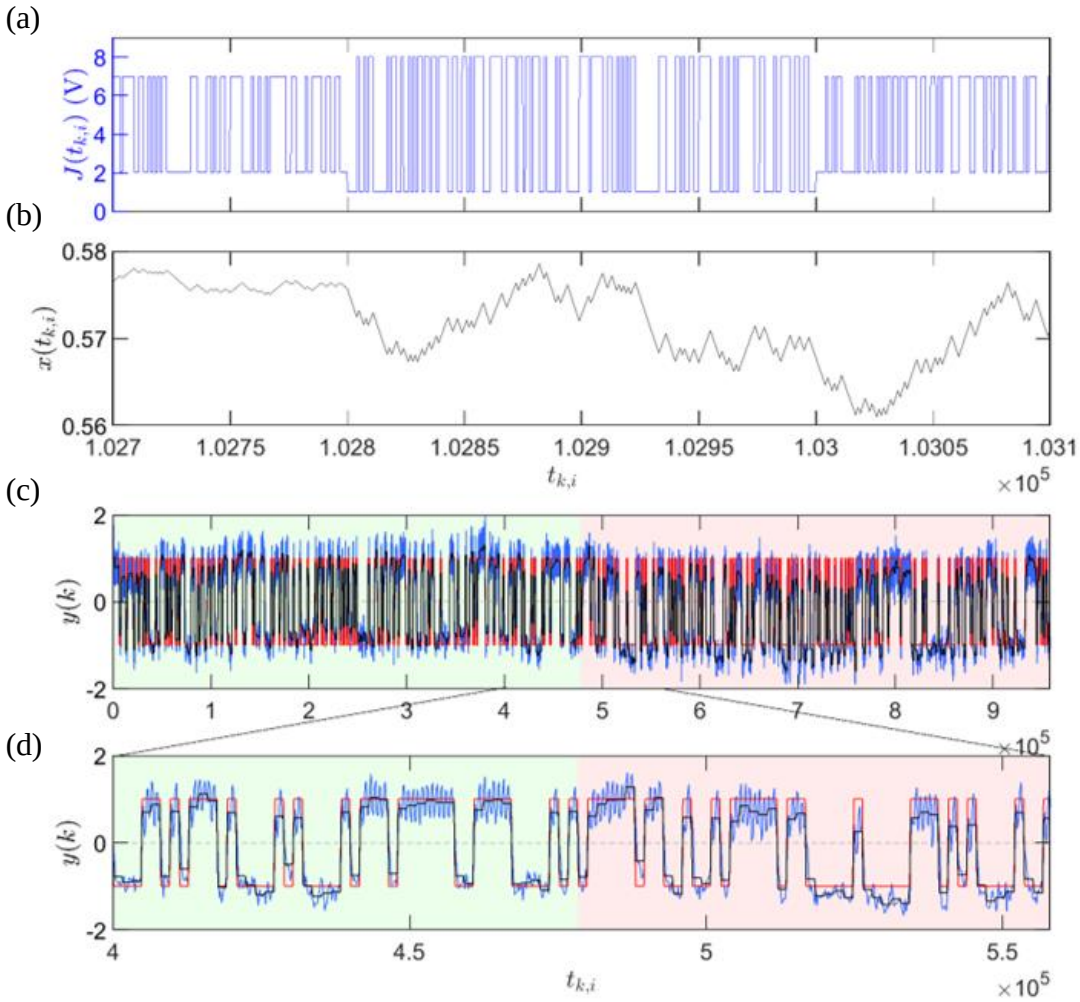


Figure C.1: Optimal result from the waveform discrimination task *with* partial delayed feedback. (a) A subsection of the input sequence $J(t)$. (b) The tunnelling regime response of the PASN to the stimulus shown in (a) using a response time $T = 130\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.201 and 0.248 for the training and testing data respectively. (d) A zoomed region of (c) showing clearly that the two waveform classes have been successfully separated. The classification is made by comparing the average of $\hat{y}$ over each waveform (black) with the decision boundary at zero.

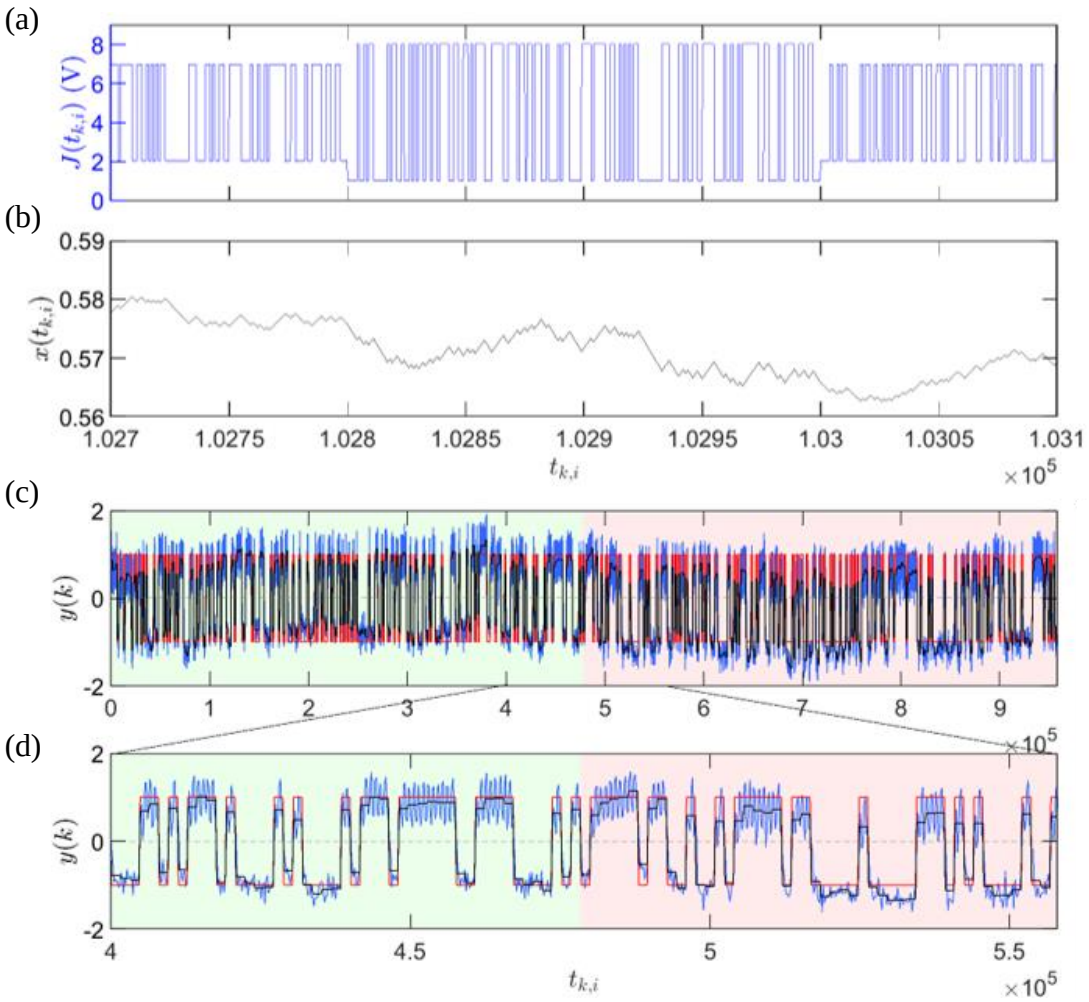


Figure C.2: Optimal result from the waveform discrimination task *without* partial delayed feedback. (a) A subsection of the input sequence $J(t)$. (b) The tunnelling regime response of the PASN to the stimulus shown in (a) using a response time $T = 270\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.214 and 0.257 for the training and testing data respectively. (d) A zoomed region of (c) showing clearly that the two waveform classes have been successfully separated. The classification is made by comparing the average of $\hat{y}$ over each waveform (black) with the decision boundary at zero.

C.2 NARMA10

C.2.1 Overfitting and Ridge Regression

This appendix presents the results of the NARMA10 task using the physical TRNDN when trained using ridge regression. Figure 6.6 presents the results from training via linear regression where it was found that the weights were being overfit to the training data. This results in a low training NRMSE of 0.084 and a large testing NRMSE of 0.265. Hence, re-training the weights using ridge regression was performed to attempt to reduce the overfitting and improve (decrease) the testing NRMSE. The parameters used are the same as in Figure 6.6: $N = 200$ and $T = 5\theta$. Figure C.3 (a) shows the recursive ridge parameter search used to

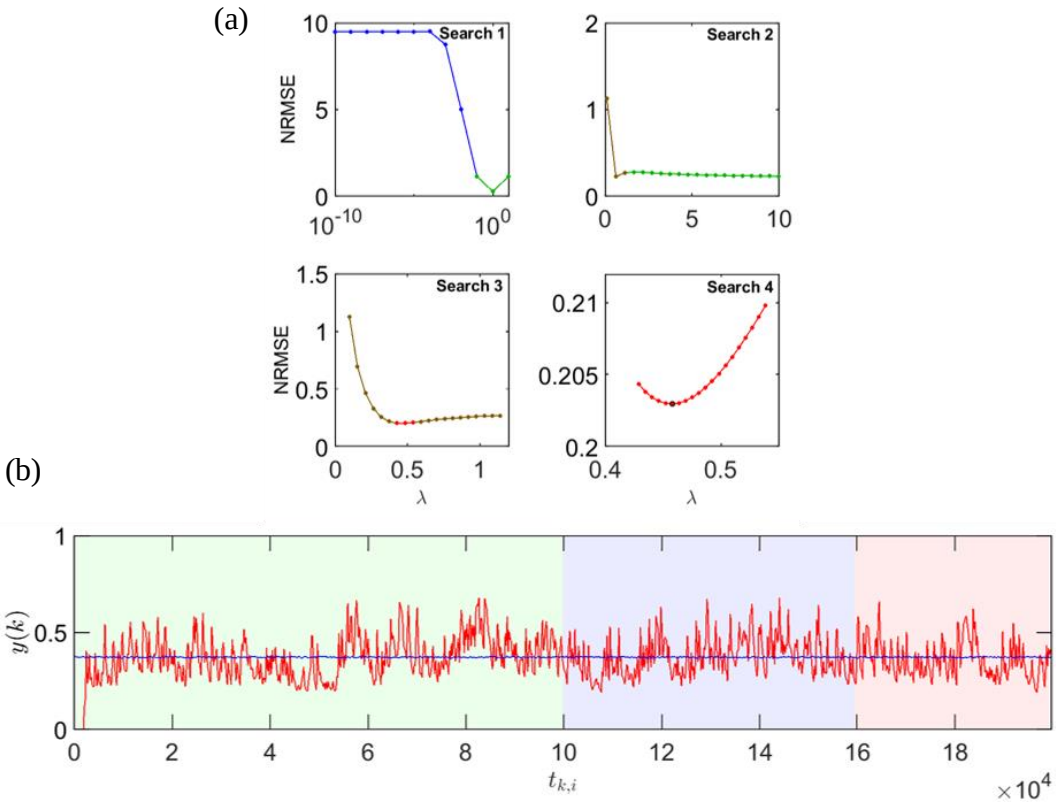


Figure C.3: NARMA10 task using ridge regression; $N = 200, T = 5$, with partial delayed feedback. (a) The recursive search used to find the optimal ridge parameter. Successive searches are performed over narrower ranges to find the minimum NRMSE for the validation data. (b) The final output of the virtual reservoir $\hat{y}$ (blue) and the NARMA10 target function (red). The result is characterised by NRMSE values of 0.175 for the training data (green background), 0.203 for the validation data (blue background), and 0.206 for the testing data (red background). The difference between the training and testing NRMSE values is significantly reduced by training with ridge regression, but $\hat{y}$ is a poor approximation of the target function.

find the optimal ridge parameter value $\lambda = 0.458$. Figure C.3 (b) shows the result of the NARMA10 task when using the weights corresponding to the optimal ridge parameter. The light green, blue and pink backgrounds represent the training, validation and testing data respectively. In this case, the weights could not be successfully trained using ridge regression. The only way that the algorithm could minimize the NRMSE was by reducing the magnitude of all of the weights, thus producing the trivial $\hat{y}$ shown in blue which is approximately a flat horizontal line. In one way, the ridge regression has served its purpose: the overfitting has been reduced, as signified by the increase in training NRMSE (0.084 to 0.175) and concurrent decrease in testing NRMSE (0.265 to 0.206). However, $\hat{y}$ is a poor approximation of the target function and so training the weights via ridge regression is deemed to be unsuccessful.

C.2.2 Results Using Optimised Time Constants

This appendix contains the results of the NARMA10 task using the physical TRNDN with the optimal time constants found in Section 6.1.2. Figure C.4 shows the result with partial delayed feedback and Figure C.5 shows the result without partial delayed feedback. These are important results to show because they represent the best performing neuromorphic implementations of real PASN devices. However, because Figure C.4 and Figure C.5 appear superficially similar to other figures in Section 6.1.2, they are included in this appendix to aid readability.

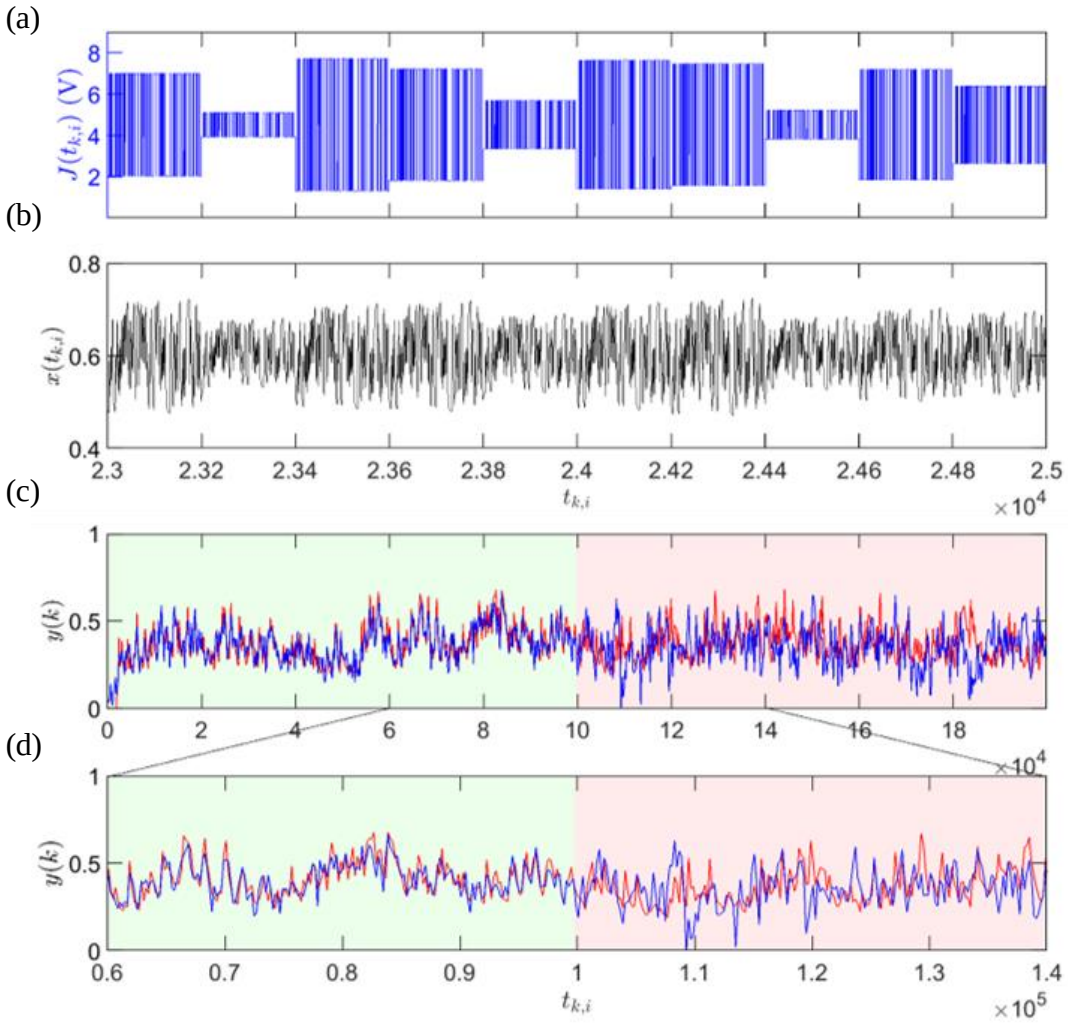


Figure C.4: Optimal result from the NARMA10 task *with* partial delayed feedback. (a) A subsection of the input sequence $J(t)$. (b) The tunnelling regime response of the PASN to the stimulus shown in (a) using a response time $T = 2\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.082 and 0.264 for the training and testing data respectively. (d) A zoomed region of (c) showing $\hat{y}$ and y in more detail. $\hat{y}$ is a good approximation of y for the training data, but less so for the testing data: a sign of overfitting.

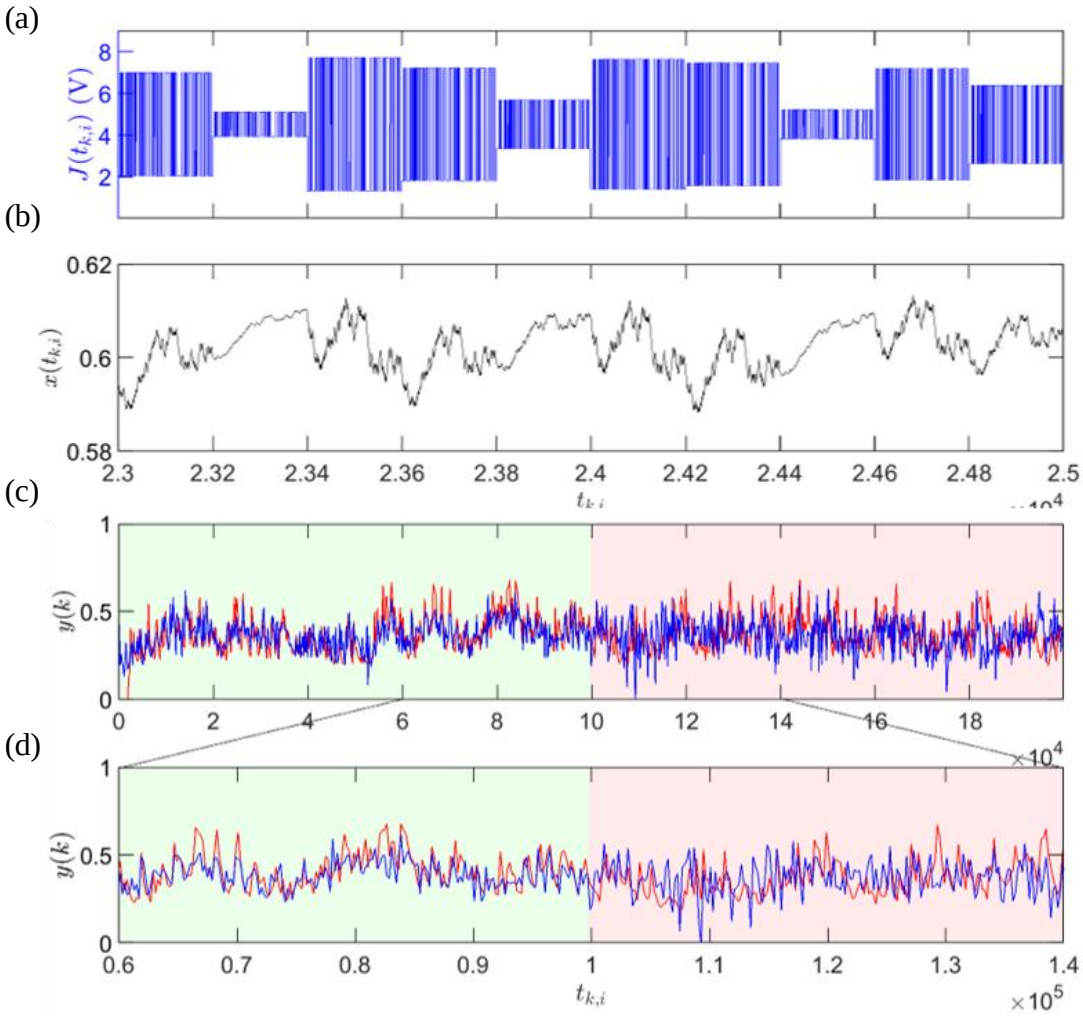


Figure C.5: Optimal result from the NARMA10 task *without* partial delayed feedback. (a) A subsection of the input sequence $J(t)$. (b) The tunnelling regime response of the PASN to the stimulus shown in (a) using a response time $T = 140\theta$. (c) The final output of the virtual reservoir $\hat{y}$ (blue) approximates the target function y (red) with NRMSE values of 0.123 and 0.253 for the training and testing data respectively. (d) A zoomed region of (c) showing $\hat{y}$ and y in more detail. $\hat{y}$ is a good approximation of y for the training data, but less so for the testing data: a sign of overfitting.

List of Acronyms

Due to the frequent use of acronyms throughout this thesis, the following list is provided as a quick reference for the reader. Note that only those acronyms frequently used are included.

- **ACF** – Autocorrelation Function: The correlation of a signal with a delayed copy of itself as a function of delay (Section 2.3.3).
- **ANN** – Artificial Neural Network: Brain-inspired computing frameworks implemented in software (Section 1.2).
- **BIC** – Bayesian Information Criterion: a criterion for model selection among a finite set of models (Section 2.3.4.4).
- **CCL** – Creative Commons Licence: A public copyright license that enables the free distribution of an otherwise copyrighted work.
- **CDF** – Cumulative Density Function: Mathematical function describing the probability that a quantity takes a value less than or equal to its argument (Section 2.3.2).
- **CMOS** – Complementary Metal-Oxide Semiconductor: The material used to fabricate circuitry for conventional computing architectures (Section 1.3).
- **DC** – Direct Current: An electrical current which flows in one direction.
- **EFISD/EFIE** – Electric Field Induced Surface Diffusion/Electric Field Induced Evaporation: Atomic scale processes which result in filament formation in PASNs (Section 1.4.5.1).
- **IC** – Integrated Circuit: A set of electronic circuits on one small flat piece of CMOS material (Section 1).
- **IEI** – Inter-Event Interval: The time between two consecutive events (Section 3.1.1).
- **IGA** – Inert Gas Aggregation: The method of producing metal clusters used in this thesis (Section 1.4.1).
- **KS Test** – Kolmogorov–Smirnov Test: A nonparametric test of the equality of continuous, one-dimensional probability distributions (Section 2.3.4.3).

- **LFP** – Local Field Potential: The electrical signal measured using an MEA (Section 3.1.2.1).
- **LRTC** – Long-Range Temporal Correlation: A property of a system which has memory characterised by and ACF which decays as a power law (Section 2.3.3).
- **MEA** – Multi-Electrode Array: A substrate featuring an array of planar electrodes used to record electrical signals from neural tissue (Section 3.1.2.1).
- **MGO –** Mackey-Glass Oscillator: A mathematical delayed feedback model capable of producing chaotic dynamics (Section 5.1).
- **MLE** – Maximum Likelihood Estimation: A method of estimating the parameters of a probability distribution (Section 2.3.4.2).
- **NARMA –** Nonlinear Auto-Regressive Moving Average: A time series prediction task used to benchmark artificial neural systems (Section 4.2.4).
- **NDN –** Nonlinear Dynamical Node: The object which emulates a virtual reservoir in TDRC (Section 4.3.2).
- **NP** – Nanoparticle: Nanoscale structures of dimension 1 – 100 nm (Section 1.4.1).
- **NRMSE –** Normalized Root-Mean Squared Error: A measure of the differences between values predicted by a model and the values observed (Section 4.2.2).
- **PASN -** Percolating Atomic Switch Network: The systems which are the focus of this thesis (Section 1.4.3.2).
- **PDF** – Probability Density Function: Mathematical function describing the distribution of relative likelihoods that a variable takes on a certain value (Section 2.3.2).
- **Physical TRNDN –** An experimental implementation of the TRNDN model which utilizes a real PASN in the tunnelling regime (Section 6.1).
- **RC –** Reservoir Computing: A RNN model in which only the output connections are trained (Section 1.2.3).
- **RNN** – Recurrent Neural Network: An ANN model which includes recurrent connections which provide memory (Section 1.2.2).
- **SLP/MLP** – Single Layer Perceptron/Multi-Layer Perceptron: Simple feed-forward ANN models (Section 1.2.1).
- **SRNDN –** Switching Regime Nonlinear Dynamical Node: An NDN model which simulates a PASN in the switching regime (Section 5.3).

- **TDRC –** Time-Delay Reservoir Computing: A variant of reservoir computing which utilizes a single nonlinear dynamical node to emulate a virtual reservoir of nonlinearly interacting virtual nodes (Section 4.1).
- **TRNDN –** Tunnelling Regime Nonlinear Dynamical Node: An NDN model which simulates a PASN in the tunnelling regime (Section 5.2).
- **WER –** Word Error Rate: A measure of the inaccuracy in spoken or written word recognition tasks (Section 4.2.5)

Bibliography

[1] J. Bardeen and W. H. Brattain, Physical Review **74**, 230 (1948).

[2] J. S. Kilby, IEEE Solid-State Circuits Society Newsletter **12**, 44 (2007).

[3] G. E. Moore, Electronics **38** (1965).

[4] D. J. Frank, IBM Journal of Research and Development **46**, 235 (2002).

[5] S. K. Moore, IEEE Spectrum **56**, 9 (2019).

[6] T. Yuan *et al.*, Proc. IEEE **85**, 486 (1997).

[7] F.-Q. Xie, L. Nittler, C. Obermair, and T. Schimmel, Physical review letters **93**, 128303 (2004).

[8] G. E. Moore, *Gordon Moore: The Man Whose Name Means Progress, the Visionary Engineer Reflects on 50 Years of Moore's Law*, edited by R. Courtland, IEEE Spectrum (2015).

[9] C. Mead, Proc. IEEE **78**, 1629 (1990).

[10] R. A. Nawrocki, R. M. Voyles, and S. E. Shaheen, IEEE Transactions on Electron Devices **63**, 3819 (2016).

[11] T. Roska, IEEE Circuits and Systems Magazine **5**, 5 (2005).

[12] C. D. Schuman, T. E. Potok, R. M. Patton, J. D. Birdwell, M. E. Dean, G. S. Rose, and J. S. Plank, arXiv preprint:1705.06963 (2017).

[13] N. Srinivasa, N. D. Stepp, and J. Cruz-Albrecht, Frontiers in neuroscience **9**, 449 (2015).

[14] S. Bose, S. Shirai, J. Mallinson, S. Acharya, E. Galli, and S. Brown, *2018 IEEE 18th International Conference on Nanotechnology (IEEE-NANO)*, (IEEE), pp. 1 (2018).

[15] S. Bose, S. Shirai, J. Mallinson, and S. Brown, Faraday Discussions **213**, 471 (2019).

[16] S. K. Bose, J. B. Mallinson, R. M. Gazoni, and S. A. Brown, IEEE Transactions on Electron Devices **64**, 5194 (2017).

[17] S. Fostner, R. Brown, J. Carr, and S. A. Brown, Physical Review B **89**, 075402 (2014).

[18] S. Fostner and S. A. Brown, Physical Review E **92**, 052134 (2015).

[19] A. Sattar, S. Fostner, and S. A. Brown, Physical Review Letters **111**, 136808 (2013).

[20] J. Park, The Yale Journal of Biology and Medicine **75**, 63 (2002).

[21] E. R. Kandel, J. H. Schwartz, T. M. Jessell, D. o. Biochemistry, M. B. T. Jessell, S. Siegelbaum, and A. Hudspeth, *Principles of Neural Science*, McGraw-Hill, New York (2000).

[22] G. M. Shepherd, *Introduction to Synaptic Circuits*, Oxford University Press (1990).

[23] M. A. Lynch, Physiological reviews **84**, 87 (2004).

[24] R. S. Frackowiak, *Human Brain Function*, Elsevier (2004).

[25] R. Yuste and G. M. Church, Scientific American **310**, 38 (2014).

[26] E. Bullmore and O. Sporns, Nature Reviews Neuroscience **13**, 336 (2012).

[27] H.-J. Park and K. Friston, Science **342**, 1238411 (2013).

[28] A. Di Ieva, *The Fractal Geometry of the Brain*, Springer (2016).

[29] B. J. He, Trends in cognitive sciences **18**, 480 (2014).

[30] J. M. Beggs and D. Plenz, Journal of neuroscience **23**, 11167 (2003).

[31] N. Friedman, S. Ito, B. A. Brinkman, M. Shimono, R. L. DeVille, K. A. Dahmen, J. M. Beggs, and T. C. Butler, Physical review letters **108**, 208102 (2012).

[32] W. L. Shew, W. P. Clawson, J. Pobst, Y. Karimipanah, N. C. Wright, and R. Wessel, Nature Physics **11**, 659 (2015).

[33] W. L. Shew and D. Plenz, The neuroscientist **19**, 88 (2013).

[34] L. Cocchi, L. L. Gollo, A. Zalesky, and M. Breakspear, Progress in neurobiology, 132 (2017).

[35] O. Kinouchi and M. Copelli, Nature physics **2**, 348 (2006).

[36] M. A. Munoz, Reviews of Modern Physics **90**, 031001 (2018).

[37] M. H. Hassoun, *Fundamentals of Artificial Neural Networks*, MIT press (1995).

[38] A. K. Jain, J. Mao, and K. M. Mohiuddin, Computer **29**, 31 (1996).

[39] I. A. Basheer and M. Hajmeer, Journal of microbiological methods **43**, 3 (2000).

[40] F. Rosenblatt, *Principles of Neurodynamics. Perceptrons and the Theory of Brain Mechanisms*, Cornell Aeronautical Lab Inc Buffalo NY (1961).

[41] R. Hecht-Nielsen, *Neurocomputer Applications*, Springer (1989).

[42] G. Cybenko, Mathematics of control, signals and systems **2**, 303 (1989).

[43] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, Neural networks **6**, 861 (1993).

[44] S. Hochreiter, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems **6**, 107 (1998).

[45] B. C. Cetin, J. W. Burdick, and J. Barhen, *IEEE International Conference on Neural Networks*, (IEEE), pp. 836 (1993).

[46] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Cognitive modeling **5**, 1 (1988).

[47] B. Karlik and A. V. Olgac, International Journal of Artificial Intelligence and Expert Systems **1**, 111 (2011).

[48] W. Wardah, M. G. Khan, A. Sharma, and M. A. Rashid, Computational Biology and Chemistry **81**, 1 (2019).

[49] Y. Bengio, Foundations and Trends in Machine Learning **2**, 1 (2009).

[50] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT press (2016).

[51] K. Vandoorne, P. Mechet, T. Van Vaerenbergh, M. Fiers, G. Morthier, D. Verstraeten, B. Schrauwen, J. Dambre, and P. Bienstman, Nature Communications **5**, 3541 (2014).

[52] F. Wyffels, B. Schrauwen, and D. Stroobandt, *International conference on artificial neural networks*, (Springer), pp. 808 (2008).

[53] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *Advances in neural information processing systems*, pp. 1097 (2012).

[54] W. Rawat and Z. Wang, Neural computation **29**, 2352 (2017).

[55] Z. C. Lipton, J. Berkowitz, and C. Elkan, arXiv preprint:1506.00019 (2015).

[56] J. T. Connor, R. D. Martin, and L. E. Atlas, IEEE transactions on neural networks **5**, 240 (1994).

[57] S. Lai, L. Xu, K. Liu, and J. Zhao, *Twenty-ninth AAAI conference on artificial intelligence*, (2015).

[58] H. Jaeger, Bonn, Germany: German National Research Center for Information Technology GMD Technical Report **148**, 13 (2001).

[59] A. Mittal, Understanding RNN and LSTM, (Towards Data Science) https://towardsdatascience.com/understanding-rnn-and-lstm-f7cdf6dfc14e (2019).

[60] B. Schrauwen, D. Verstraeten, and J. Van Campenhout, *Proceedings of the 15th european symposium on artificial neural networks. p. 471-482 2007*, pp. 471 (2007).

[61] W. Maass, T. Natschläger, and H. Markram, Neural computation **14**, 2531 (2002).

[62] Y. Bengio, A. Courville, and P. Vincent, IEEE Transactions on Pattern Analysis and Machine Intelligence **35**, 1798 (2013).

[63] J. Schmidhuber, Neural networks **61**, 85 (2015).

[64] Y. LeCun, Y. Bengio, and G. Hinton, nature **521**, 436 (2015).

[65] H. Jaeger, Scholarpedia **2**, 2330 (2007).

[66] K. Scharnhorst, W. Woods, C. Teuscher, A. Stieg, and J. Gimzewski, *2017 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*, (IEEE), pp. 133 (2017).

[67] H. O. Sillin, R. Aguilera, H.-H. Shieh, A. V. Avizienis, M. Aono, A. Z. Stieg, and J. K. Gimzewski, Nanotechnology **24**, 384004 (2013).

[68] C. Mesaritakis, V. Papataxiarhis, and D. Syvridis, JOSA B **30**, 3048 (2013).

[69] J. C. Coulombe, M. C. York, and J. Sylvestre, PloS one **12**, e0178663 (2017).

[70] M. Dale, J. F. Miller, S. Stepney, and M. A. Trefzer, *International Conference on Unconventional Computation and Natural Computation*, (Springer), pp. 49 (2016).

[71] J. Bürger, A. Goudarzi, D. Stefanovic, and C. Teuscher, *Nanoscale Architectures (NANOARCH), 2015 IEEE/ACM International Symposium on*, (IEEE), pp. 33 (2015).

[72] J. Von Neumann, IEEE Annals of the History of Computing **15**, 27 (1993).

[73] J. Backus, *Can Programming Be Liberated from the Von Neumann Style?: A Functional Style and Its Algebra of Programs*, ACM (2007).

[74] E. Kandel and S. Siegelbaum, *Principles of Neural Science*, Elsevier (1985).

[75] D. Fox, *Brain-Like Chip May Solve Computers Big Problem: Energy* (2009).

[76] M. Davies *et al.*, IEEE Micro **38**, 82 (2018).

[77] N. Qiao, H. Mostafa, F. Corradi, M. Osswald, F. Stefanini, D. Sumislawska, and G. Indiveri, Frontiers in neuroscience **9**, 141 (2015).

[78] T. Fukuda, T. Shibata, M. Tokita, and T. Mitsuoka, IEEE Transactions on Industrial Electronics **39**, 497 (1992).

[79] F. Baletto and R. Ferrando, Reviews of modern physics **77**, 371 (2005).

[80] M. Naito, T. Yokoyama, K. Hosokawa, and K. Nogi, *Nanoparticle Technology Handbook*, Elsevier (2018).

[81] C. Buzea, I. I. Pacheco, and K. Robbie, Biointerphases **2**, MR17 (2007).

[82] X. Batlle and A. l. Labarta, Journal of Physics D: Applied Physics **35**, R15 (2002).

[83] P. Buffat and J. P. Borel, Physical review A **13**, 2287 (1976).

[84] M.-C. Daniel and D. Astruc, Chemical reviews **104**, 293 (2004).

[85] R. A. Taylor, T. Otanicar, and G. Rosengarten, Light: Science & Applications **1**, e34 (2012).

[86] R. Singh and H. S. Nalwa, Journal of biomedical nanotechnology **7**, 489 (2011).

[87] J. W. Wiechers and N. Musee, Journal of biomedical nanotechnology **6**, 408 (2010).

[88] J. Partridge, R. Reichel, A. Ayesh, D. Mackenzie, and S. Brown, physica status solidi (a) **203**, 1217 (2006).

[89] R. Reichel, J. G. Partridge, A. D. Dunbar, S. A. Brown, O. Caughley, and A. Ayesh, Journal of Nanoparticle Research **8**, 405 (2006).

[90] S. Arcidiacono, N. Bieri, D. Poulikakos, and C. Grigoropoulos, International Journal of Multiphase Flow **30**, 979 (2004).

[91] M. Jose-Yacaman, C. Gutierrez-Wing, M. Miki, D.-Q. Yang, K. Piyakis, and E. Sacher, The Journal of Physical Chemistry B **109**, 9703 (2005).

[92] D. Moldovan, V. Yamakov, D. Wolf, and S. R. Phillpot, Physical review letters **89**, 206101 (2002).

[93] A. Sattar, Electrical Characterization of Cluster Devices, PhD Thesis, University of Canterbury (2011).

[94] S. R. Broadbent and J. M. Hammersley, *Mathematical Proceedings of the Cambridge Philosophical Society*, (Cambridge University Press), pp. 629 (1957).

[95] D. Stauffer and A. Aharony, *Introduction to Percolation Theory: Revised Second Edition*, CRC press (2014).

[96] S. Kirkpatrick, Reviews of modern physics **45**, 574 (1973).

[97] J. Mallinson, S. Shirai, S. Acharya, S. Bose, E. Galli, and S. Brown, Science advances **5**, eaaw8438 (2019).

[98] R. M. Hill, Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences **309**, 377 (1969).

[99] S. Feng, B. Halperin, and P. Sen, Physical Review B **35**, 197 (1987).

[100] J. J. Sakurai and E. D. Commins, *Modern Quantum Mechanics, Revised Edition*, American Association of Physics Teachers (1995).

[101] A. B. Kaiser, Reports on Progress in Physics **64**, 1 (2001).

[102] J. Simmons, Journal of Physics D: Applied Physics **4**, 613 (1971).

[103] C. Xiang, J. Y. Kim, and R. M. Penner, Nano letters **9**, 2133 (2009).

[104] M. Olsen, M. Hummelgård, and H. Olin, PloS one **7**, e30106 (2012).

[105] T. Ganetsos, A. Mair, G. Mair, L. Bischoff, C. Akhmadaliev, and C. Aidinis, Surface and interface analysis **39**, 128 (2007).

[106] J. Homoth *et al.*, Nano letters **9**, 1588 (2009).

[107] M. Trouwborst, S. Van Der Molen, and B. Van Wees, Journal of Applied Physics **99**, 114316 (2006).

[108] H. Ohno and Y. Katano, *Materials Science Forum*, (Trans Tech Publ), pp. 215 (1989).

[109] G. Audi, O. Bersillon, J. Blachot, and A. H. Wapstra, Nuclear Physics A **624**, 1 (2003).

[110] M. Varsanyi, J. Jaen, A. Vertes, and L. Kiss, Electrochimica Acta **30**, 529 (1985).

[111] S. Cho, J. Yu, S. K. Kang, and D. Shih, JOM **57**, 50 (2005).

[112] E. Sutter, F. Ivars-Barcelo, and P. Sutter, Particle & Particle Systems Characterization **31**, 879 (2014).

[113] R. Reichel, Nano Scale Cluster Devices, PhD Thesis, University of Canterbury (2007).

[114] C. B. Nakhosteen and K. Jousten, *Handbook of Vacuum Technology*, John Wiley & Sons (2016).

[115] J. F. O'Hanlon, *A User's Guide to Vacuum Technology*, John Wiley & Sons (2005).

[116] P. J. Kelly and R. D. Arnell, Vacuum **56**, 159 (2000).

[117] R. Behrisch and W. Eckstein, *Sputtering by Particle Bombardment: Experiments and Computer Calculations from Threshold to Mev Energies*, Springer (2007).

[118] A. I. Ayesh, Device Fabrication Using Bi Nanoclusters, PhD Thesis (2007).

[119] H. Haberland, *Clusters of Atoms and Molecules: Theory, Experiment, and Clusters of Atoms*, Springer Science & Business Media (2013).

[120] B. Von Issendorff and R. Palmer, Review of scientific instruments **70**, 4497 (1999).

[121] A. Clauset, C. R. Shalizi, and M. E. Newman, SIAM review **51**, 661 (2009).

[122] M. Karsai, K. Kaski, A.-L. Barabási, and J. Kertész, Scientific reports **2**, 397 (2012).

[123] K. Balakrishnan, *Exponential Distribution: Theory, Methods and Applications*, Routledge (2018).

[124] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*, John Wiley & Sons (2015).

[125] S. Shirai, Neuromorphic Behaviour in Nanoparticle Films, PhD Thesis (2019).

[126] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to Linear Regression Analysis*, John Wiley & Sons (2012).

[127] G. A. Seber and A. J. Lee, *Linear Regression Analysis*, John Wiley & Sons (2012).

[128] M. L. Goldstein, S. A. Morris, and G. G. Yen, The European Physical Journal B **41**, 255 (2004).

[129] S. I. Vrieze, Psychological methods **17**, 228 (2012).

[130] V. Darley, Artificial Life **4**, 411 (1994).

[131] P. C. Davies, arXiv preprint: 0408014 (2004).

[132] M. Hopkins, G. Pineda-García, P. A. Bogdan, and S. B. Furber, Interface Focus **8** (2018).

[133] P. A. Merolla *et al.*, Science **345**, 668 (2014).

[134] M. Prezioso, F. Merrikh-Bayat, B. Hoskins, G. Adam, K. K. Likharev, and D. B. Strukov, Nature **521**, 61 (2015).

[135] P. M. Sheridan, F. Cai, C. Du, W. Ma, Z. Zhang, and W. D. Lu, Nature nanotechnology **12**, 784 (2017).

[136] Z. Wang *et al.*, Nature Electronics **1**, 137 (2018).

[137] K. Terabe, T. Hasegawa, T. Nakayama, and M. Aono, Nature **433**, 47 (2005).

[138] T. Tuma, A. Pantazi, M. Le Gallo, A. Sebastian, and E. Eleftheriou, Nature nanotechnology **11**, 693 (2016).

[139] K. Linkenkaer-Hansen, V. V. Nikouline, J. M. Palva, and R. J. Ilmoniemi, Journal of Neuroscience **21**, 1370 (2001).

[140] J. M. Palva, A. Zhigalov, J. Hirvonen, O. Korhonen, K. Linkenkaer-Hansen, and S. Palva, Proceedings of the National Academy of Sciences **110**, 3585 (2013).

[141] A. Di Ieva, F. Grizzi, H. Jelinek, A. J. Pellionisz, and G. A. Losa, The Neuroscientist **20**, 403 (2014).

[142] A. Corral, Physical Review Letters **92**, 108501 (2004).

[143] M. Wheatland, P. Sturrock, and J. McTiernan, The Astrophysical Journal **509**, 448 (1998).

[144] A.-L. Barabasi, Nature **435**, 207 (2005).

[145] T. Kemuriyama, H. Ohta, Y. Sato, S. Maruyama, M. Tandai-Hiruma, K. Kato, and Y. Nishida, BioSystems **101**, 144 (2010).

[146] M. Riou *et al.*, *2017 IEEE International Electron Devices Meeting (IEDM)*, (IEEE), pp. 36.3. 1 (2017).

[147] S. Vajna, B. Tóth, and J. Kertész, New Journal of Physics **15**, 103023 (2013).

[148] B. J. Berne, J. P. Boon, and S. A. Rice, The Journal of Chemical Physics **45**, 1086 (1966).

[149] Y. Tsubo, Y. Isomura, and T. Fukai, PLoS computational biology **8**, e1002461 (2012).

[150] P. Bak, C. Tang, and K. Wiesenfeld, Physical review A **38**, 364 (1988).

[151] V. Frette, K. Christensen, A. Malthe-Sørenssen, J. Feder, T. Jøssang, and P. Meakin, Nature **379**, 49 (1996).

[152] R. Rammal, C. Tannous, P. Breton, and A.-M. Tremblay, Physical review letters **54**, 1718 (1985).

[153] B. B. Mandelbrot, *The Fractal Geometry of Nature*, WH freeman New York (1983).

[154] A. Sornette and D. Sornette, EPL (Europhysics Letters) **9**, 197 (1989).

[155] L. Turnbull, E. Dian, and G. Gross, Journal of neuroscience methods **145**, 23 (2005).

[156] S. Zapperi, K. B. Lauritsen, and H. E. Stanley, Physical review letters **75**, 4071 (1995).

[157] I. Bar-Gad, Y. a. Ritov, and H. Bergman, Journal of neuroscience methods **104**, 155 (2001).

[158] C. W. Eurich, J. M. Herrmann, and U. A. Ernst, Physical review E **66**, 066137 (2002).

[159] C. Haldeman and J. M. Beggs, Physical review letters **94**, 058101 (2005).

[160] T. E. Harris, *The Theory of Branching Process*, Springer-Verlag (1964).

[161] R. Peierls, *Mathematical Proceedings of the Cambridge Philosophical Society*, (Cambridge University Press), pp. 477 (1936).

[162] J. X. De Carvalho and C. P. Prado, Physical review letters **84**, 4006 (2000).

[163] D. Avnir, O. Biham, D. Lidar, and O. Malcai, Science **279**, 39 (1998).

[164] M. P. Stumpf and M. A. Porter, Science **335**, 665 (2012).

[165] M. E. Newman, Contemporary physics **46**, 323 (2005).

[166] G. A. Miller, E. Newman, and E. Friedman, American Journal of Psychology **70**, 311 (1957).

[167] W. Li, IEEE Transactions on information theory **38**, 1842 (1992).

[168] M. F. Shlesinger, G. M. Zaslavsky, and U. Frisch, (1995).

[169] H. Takayasu, Physical review letters **63**, 2563 (1989).

[170] M. Benayoun, J. D. Cowan, W. van Drongelen, and E. Wallace, PLoS computational biology **6**, e1000846 (2010).

[171] J. P. Sethna, K. A. Dahmen, and C. R. Myers, Nature **410**, 242 (2001).

[172] O. Perkovic, K. A. Dahmen, and J. P. Sethna, arXiv preprint: 9609072 (1996).

[173] D. Spasojević, S. Bukvić, S. Milošević, and H. E. Stanley, Physical Review E **54**, 2531 (1996).

[174] A. P. Mehta, A. C. Mills, K. A. Dahmen, and J. P. Sethna, Physical Review E **65**, 046139 (2002).

[175] J. Touboul and A. Destexhe, Physical Review E **95**, 012413 (2017).

[176] H. J. Jensen, *Self-Organized Criticality: Emergent Complex Behavior in Physical and Biological Systems*, Cambridge university press (1998).

[177] T. Petermann, T. C. Thiagarajan, M. A. Lebedev, M. A. Nicolelis, D. R. Chialvo, and D. Plenz, Proceedings of the National Academy of Sciences **106**, 15921 (2009).

[178] G. Hahn, T. Petermann, M. N. Havenith, S. Yu, W. Singer, D. Plenz, and D. Nikolić, Journal of neurophysiology **104**, 3312 (2010).

[179] A. Ponce-Alvarez, A. Jouary, M. Privat, G. Deco, and G. Sumbre, Neuron **100**, 1446 (2018).

[180] G. Scott, E. D. Fagerholm, H. Mutoh, R. Leech, D. J. Sharp, W. L. Shew, and T. Knöpfel, Journal of Neuroscience **34**, 16611 (2014).

[181] J. M. Beggs and N. Timme, Frontiers in physiology **3**, 163 (2012).

[182] N. Dehghani, N. G. Hatsopoulos, Z. D. Haga, R. Parker, B. Greger, E. Halgren, S. S. Cash, and A. Destexhe, Frontiers in physiology **3**, 302 (2012).

[183] V. Priesemann, M. Wibral, M. Valderrama, R. Pröpper, M. Le Van Quyen, T. Geisel, J. Triesch, D. Nikolić, and M. H. Munk, Frontiers in systems neuroscience **8**, 108 (2014).

[184] D. R. Chialvo, Nature physics **6**, 744 (2010).

[185] D. R. Chialvo, Nature physics **2**, 301 (2006).

[186] J. Hesse and T. Gross, Frontiers in systems neuroscience **8**, 166 (2014).

[187] P. Moretti and M. A. Muñoz, Nature communications **4**, 2521 (2013).

[188] P. Massobrio, L. de Arcangelis, V. Pasquale, H. J. Jensen, and D. Plenz, Frontiers in systems neuroscience **9**, 22 (2015).

[189] V. M. Eguiluz, D. R. Chialvo, G. A. Cecchi, M. Baliki, and A. V. Apkarian, Physical review letters **94**, 018102 (2005).

[190] W. L. Shew, H. Yang, S. Yu, R. Roy, and D. Plenz, Journal of neuroscience **31**, 55 (2011).

[191] Z. Deng and Y. Zhang, IEEE Transactions on Neural Networks **18**, 1364 (2007).

[192] H. Jaeger and H. Haas, science **304**, 78 (2004).

[193] A. Z. Stieg, A. V. Avizienis, H. O. Sillin, C. Martin-Olmos, M. Aono, and J. K. Gimzewski, Advanced Materials **24**, 286 (2012).

[194] K. Scharnhorst, Beyond Moore Neuromorphic Chips: Harnessing Complexity in Atomic Switch Networks for Alternative Computing, PhD Thesis, UCLA (2018).

[195] R. F. Voss, Journal of Physics A: Mathematical and General **17**, L373 (1984).

[196] N. Marshall, N. M. Timme, N. Bennett, M. Ripp, E. Lautzenhiser, and J. M. Beggs, Frontiers in physiology **7**, 250 (2016).

[197] B. M. McCoy and T. T. Wu, *The Two-Dimensional Ising Model*, Courier Corporation (2014).

[198] N. M. Timme, N. J. Marshall, N. Bennett, M. Ripp, E. Lautzenhiser, and J. M. Beggs, Frontiers in physiology **7**, 425 (2016).

[199] D. Rickles, P. Hawe, and A. Shiell, Journal of Epidemiology & Community Health **61**, 933 (2007).

[200] A. J. Fontenele *et al.*, Physical review letters **122**, 208101 (2019).

[201] Y. Paquot, F. Duport, A. Smerieri, J. Dambre, B. Schrauwen, M. Haelterman, and S. Massar, Scientific reports **2**, 287 (2012).

[202] L. Appeltant, M. C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. R. Mirasso, and I. Fischer, Nature communications **2**, 468 (2011).

[203] M. Lukoševičius, H. Jaeger, and B. Schrauwen, KI-Künstliche Intelligenz **26**, 365 (2012).

[204] J. Torrejon *et al.*, Nature **547**, 428 (2017).

[205] L. Appeltant, Reservoir Computing Based on Delay-Dynamical Systems, PhD Thesis, Vrije Universiteit Brussel/Universitat de les Illes Balears (2012).

[206] C. Du, F. Cai, M. A. Zidan, W. Ma, S. H. Lee, and W. D. Lu, Nature communications **8**, 2204 (2017).

[207] L. Larger, A. Baylón-Fuentes, R. Martinenghi, V. S. Udaltsov, Y. K. Chembo, and M. Jacquot, Physical Review X **7**, 011015 (2017).

[208] M. Minksy and S. Papert, MIT press **200**, 355 (1969).

[209] M. L. Alomar, M. C. Soriano, M. Escalona-Morán, V. Canals, I. Fischer, C. R. Mirasso, and J. L. Rosselló, IEEE Transactions on Circuits and Systems II: Express Briefs **62**, 977 (2015).

[210] D. Marković *et al.*, Applied Physics Letters **114**, 012409 (2019).

[211] P. Antonik, F. Duport, M. Hermans, A. Smerieri, M. Haelterman, and S. Massar, IEEE transactions on neural networks and learning systems **28**, 2686 (2016).

[212] A. F. Atiya and A. G. Parlos, IEEE transactions on neural networks **11**, 697 (2000).

[213] A. Goudarzi, M. R. Lakin, and D. Stefanovic, *International Conference on Unconventional Computation and Natural Computation*, (Springer), pp. 164 (2014).

[214] P. Nieters, J. Leugering, and G. Pipa, IBM Journal of Research and Development **61**, 8: 7 (2017).

[215] K. Bai and Y. Yi, ACM Journal on Emerging Technologies in Computing Systems (JETC) **14**, 1 (2018).

[216] W. Jiang, L. Chen, K. Zhou, L. Li, Q. Fu, Y. Du, and R. Liu, Applied Physics Letters **115**, 192403 (2019).

[217] G. R. Doddington and T. B. Schalk, IEEE spectrum **18**, 26 (1981).

[218] R. Lyon, *ICASSP'82. IEEE International Conference on Acoustics, Speech, and Signal Processing*, (IEEE), pp. 1282 (1982).

[219] F. A. Araujo *et al.*, Scientific Reports **10**, 1 (2020).

[220] U. Huebner, N. Abraham, and C. Weiss, Physical Review A **40**, 6354 (1989).

[221] Y. LeCunThe Mnist Database of Handwritten Digits, http://yann.lecun.com/exdb/mnist/ (1998).

[222] E. Kussul and T. Baidyk, Image and Vision Computing **22**, 971 (2004).

[223] P. De Chazal, J. Tapson, and A. Van Schaik, *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, (IEEE), pp. 2165 (2015).

[224] P. J. Grother, National Institute of Standards and Technology (1995).

[225] D. Ciregan, U. Meier, and J. Schmidhuber, *2012 IEEE conference on computer vision and pattern recognition*, (IEEE), pp. 3642 (2012).

[226] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, *2017 International Joint Conference on Neural Networks (IJCNN)*, (IEEE), pp. 2921 (2017).

[227] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, Frontiers in neuroscience **9**, 437 (2015).

[228] R. Wang, C. S. Thakur, G. Cohen, T. J. Hamilton, J. Tapson, and A. van Schaik, IEEE transactions on biomedical circuits and systems **11**, 574 (2017).

[229] D. Zhang, L. Zeng, Y. Qu, Z. M. Wang, W. Zhao, T. Tang, and Y. Wang, *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, (IEEE), pp. 1538 (2015).

[230] L. Appeltant, G. Van der Sande, J. Danckaert, and I. Fischer, Scientific reports **4**, 3629 (2014).

[231] I. Aleksander and H. Morton, *Proceedings of the 1992 International Conference on Artificial Neural Networks (ICANN-92)*, (North Holland), p. 863 (1991).

[232] G. Tanaka, T. Yamane, J. B. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose, Neural Networks (2019).

[233] S. Lepri, G. Giacomelli, A. Politi, and F. Arecchi, Physica D: Nonlinear Phenomena **70**, 235 (1994).

[234] D. Verstraeten, B. Schrauwen, and D. Stroobandt, *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, (IEEE), pp. 1050 (2006).

[235] L. Larger, M. C. Soriano, D. Brunner, L. Appeltant, J. M. Gutiérrez, L. Pesquera, C. R. Mirasso, and I. Fischer, Optics express **20**, 3241 (2012).

[236] K. Ikeda, Optics communications **30**, 257 (1979).

[237] M. C. Soriano, S. Ortín, D. Brunner, L. Larger, C. R. Mirasso, I. Fischer, and L. Pesquera, Optics express **21**, 12 (2013).

[238] A. Rodan and P. Tino, IEEE transactions on neural networks **22**, 131 (2010).

[239] A. Rodan and P. Tiňo, *International Conference on Intelligent Data Engineering and Automated Learning*, (Springer), pp. 267 (2010).

[240] Q. Vinckier, F. Duport, A. Smerieri, K. Vandoorne, P. Bienstman, M. Haelterman, and S. Massar, Optica **2**, 438 (2015).

[241] M. Riou *et al.*, Physical Review Applied **12**, 024049 (2019).

[242] R. Martinenghi, S. Rybalko, M. Jacquot, Y. K. Chembo, and L. Larger, Physical review letters **108**, 244101 (2012).

[243] D. Brunner, M. C. Soriano, C. R. Mirasso, and I. Fischer, Nature communications **4**, 1 (2013).

[244] M. C. Soriano, S. Ortín, L. Keuninckx, L. Appeltant, J. Danckaert, L. Pesquera, and G. Van der Sande, IEEE transactions on neural networks and learning systems **26**, 388 (2014).

[245] H. Jaeger, *Advances in neural information processing systems*, pp. 609 (2003).

[246] M. C. Mackey and L. Glass, Science **197**, 287 (1977).

[247] A. N. Tait, T. F. De Lima, E. Zhou, A. X. Wu, M. A. Nahmias, B. J. Shastri, and P. R. Prucnal, Scientific reports **7**, 1 (2017).

[248] X. Dutoit, B. Schrauwen, J. Van Campenhout, D. Stroobandt, H. Van Brussel, and M. Nuttin, Neurocomputing **72**, 1534 (2009).